ASPLinux

Руководство пользователя

Оглавление

Глава 1

Введение

Цель настоящего руководства — дать представление о практическом использовании **ASPLinux**, в первую очередь — как настольной системы для конечного пользователя. Предполагается, что читатель имеет представление об основных понятиях Linux в объеме руководства *«Быстрый старт»* и о настройке системы в аспектах, важных для конечного пользователя, в объеме *«Руководства по установке и начальной настройке»*. В то же время детальных знаний об управлении системой и ее тонких настройках не требуется — сведения этого рода сконцентрированы в *«Руководстве администратора»*, которое может изучаться после данного руководства (или параллельно с ним).

С позиций конечного пользователя, программы, входящие в любую операционную систему (и **ASPLinux** здесь не исключение) можно разделить на три основные группы:

- рабочие среды, предназначенные для организации комфортной работы пользователя;
- пользовательские утилиты;
- прикладные пакеты, посредством которых создаются пользовательские данные.

В соответствии с этим, руководство неявным образом разбивается на три части.

Первая часть посвящена приемам использования рабочих сред текстового (командные оболочки) и графического (интегрированные среды, оконные менеджеры) режимов. Она предваряется описанием способов получения дополнительной информации внутренними средствами Linux, то есть экранной документации.

Во второй части рассматриваются пользовательские утилиты, служащие для управления данными. Это, в первую очередь, средства работы с файлами,

включая инструменты для резервного копирования, и во вторую — средства для работы в Интернет (браузеры, почтовые и ftp-клиенты и т.д.).

Наконец, третья часть включает описание прикладных пакетов: текстовых редакторов, средств для автоматизации офисной работы, графических редакторов и мультимедийных приложений.

Разумеется, в рамках одного руководства невозможно с равной степенью полноты описать все изобилие пакетов, входящих в состав дистрибутива **ASPLinux**. Поэтому внимание пользователя концентрируется на наиболее распространенных и наиболее эффективных средствах в каждой из обозначенных выше групп. Для более полного знакомства с пользовательскими программами Linux следует обратиться к дополнительным источникам информации, обзор которых приведен в заключении.

1.1 Информация для читателей

В случае, если Вы заметили опечатку в этой книге или у Вас есть предложения по улучшению её содержания, пожалуйста, оправьте электронное письмо по адресу `support@asplinux.ru` с пометкой «Документация» или оставьте свой комментарий в специальной системе отслеживания ошибок Bugzilla по адресу `http://bugzilla.asplinux.ru/`

Глава 2

Методы работы с документацией

Ни одно руководство по Linux, сколь бы полным и объемным оно ни было, не в состоянии охватить все многообразие команд и пакетов этой системы и всех способов их применения. И потому первое, чему должен научиться пользователь Linux, вне зависимости от стоящих перед ним задач, — это методам получения дополнительной информации по интересующим его вопросам.

Для этого служат системы интерактивого руководства — отдаленный аналог системы помощи в MS DOS/Windows, — созданные многолетними усилиями пользователей и разработчиков UNIX и UNIX-подобных систем. Вне зависимости от авторства, все они англоязычны. Однако многие документы переведены на другие языки. В частности, в дистрибутив **ASPLinux** включено большое количество документации, переведенной на русский язык. И если система была установлена в варианте русской локализации, при обращении к интерактивному руководству вызываются именно русскоязычные версии (если они существуют).

2.1 Экранная документация Manual Pages

Основным источником получения справочной информации являются страницы руководства `Manual Pages` (сокращенно — man-страницы, или просто страницы). Они вызываются следующим образом:

```
man имя_команды (или имя_программы)
```

Каждая страница начинается с имени команды (или программы) и указания на номер группы страниц, к которым она отнесена (что такое группа страниц, будет сказано чуть ниже). Далее следуют название команды с ее синонимами, если таковые имеются, и краткая справка по использованию. Например, для команды `ls` начало man-страницы в русскоязычном варианте выглядит следующим образом:

```
LS(1)                                                        LS(1)

НАЗВАНИЕ
       ls,  dir,  vdir -  программы  для отображения содержимого
       каталога

СИНТАКСИС
       ls [опции] [файл...]
       dir [файл...]
       vdir [файл...]

       Опции POSIX: [-CFRacdilqrtu1] [--]

       Опции GNU (краткая форма):  [-1abcdfghiklmnopqrstuvwxABCD-
       FGHLNQRSUX]  [-w cols] [-T cols] [-I шаблон] [--full-time]
       [--show-control-chars]    [--block-size=размер]    [--for-
       mat={long,verbose,commas,across,vertical,single-column}]
       [--sort={none,time,size,extension}]
       [--time={atime,access,use,ctime,status}]
       [--color[={none,auto,always}]] [--help] [--version] [--]
```

и так далее. Затем следует более или менее подробное описание использования команды:

```
ОПИСАНИЕ
       Программа  ls  сначала  выводит  список  всех  файлов  (не
       каталогов),  перечисленных  в  командной  строке,  а затем
       выводит список всех файлов, находящихся в каталогах.
```

включая ее опции:

```
ОПЦИИ POSIX
       -C     Печатает список файлов в колонках  (с  вертикальной
              сортировкой).
```

В заключение, как правило, даются сведения о найденных ошибках (Bug Report) и приведен список страниц, тематически связанных с данной, с указанием номера группы, к которой они принадлежат.

Большинство страниц занимает более одного экрана. В этом случае просмотр их осуществляется либо построчно (нажатием клавиши `Enter` — вперед, клавиш `Down` и `Up` — вперед и назад), либо постранично (клавишей `Пробел` — вперед, клавишами `PageDown` и `PageUp` — соответственно, на страницу вперед и назад). Для выхода из режима просмотра страницы предусмотрена клавиша `q`.

Местоположение man-страниц — в каталоге /usr/share/man; отдельные страницы man расположены также в каталогах /usr/man, /usr/local/man, /usr/X11R6/man.

Страницы разделены на восемь нумерованных групп, каждая из которых размещена в собственном подкаталоге:

```
/usr/share/man/man1
/usr/share/man/man2
```

и т.д.

Назначение этих групп следующее:

1. команды и прикладные программы пользователя,
2. системные вызовы,
3. библиотечные функции,
4. специальные файлы,
5. форматы файлов и соглашения,
6. игры,
7. макропакеты,
8. команды администрирования системы.

Наибольший интерес для пользователя представляют разделы 1 и 5, хотя, как показано в *«Руководстве администратора»*, ему нередко придется обращаться и к разделу 8. Заметим, что разделы 2-4 предназначены в основном для разработчиков программного обеспечения.

В состав разных групп могут входить разные страницы с одним и тем же именем. Так, например, в группе 1 имеется страница `tty(1)`, посвященная одноименной команде, а в группе 4 — страница `tty(4)`, описывающая драйвер для устройства `/dev/tty`. Именно поэтому в первой строке каждой страницы указывается ее принадлежность в группе. Для вызова определенной группы последняя должна быть задана в явном виде:

```
man 1 tty
```

или

```
man 4 tty
```

При вводе команды `man` без номера группы имя соответствующей страницы ищется в первую очередь в группе 1, затем в группе 8, и затем — во всех остальных в порядке возрастания номеров.

Локализованные версии страниц интерактивного руководства также расположены в каталоге /usr/share/man в собственных подкаталогах, например: de_DE — страницы руководства на немецком языке для Германии, fr_FR — франкоязычные страницы для Франции, ru — русскоязычные страницы. В данных подкаталогах страницы также разделены на группы, идентичные главной иерархии:

```
/usr/share/man/ru/man1
/usr/share/man/ru/man2
```

При корректной установке национальной локали, например, русской (а в дистрибутиве **ASPLinux** есть возможность выбора рабочей локали на стадии установки системы. см. *«Руководство по установке»*), вызов определенной страницы приводит к замене оригинальной версии соответствующей национальной (в нашем случае — русскоязычной), если последняя имеется.

Обращение к страницам руководства позволяет получить практически исчерпывающую информацию по любым командам **ASPLinux**, но только в том случае, если пользователь знает название команды, требующейся в данном случае. Однако, можно прибегнуть к поиску страниц по ключевым словам, отражающим ее функции. Например, команда

```
man -k print
```

выведет на экран список всех страниц руководства, описывающих команды, имеющие отношение к печати:

```
arch (1) - print machine architecture
banner (6) - print large banner on printer
cancel (1) - send cancel requests to an LPRng print service
```

и т.д. (список может быть очень длинным) — с номерами их групп и краткими пояснениями. Поиск по ключевым словам ведется в базе данных, которая хранится в файле /usr/share/man/whatis. Обновление базы данных производится автоматически раз в сутки.

Исчерпывающим руководством по использованию системы Manual Pages является ее собственная man-страница. Доступ к ней осуществляется по команде

```
man man
```

которая выводит на экран описание этой команды.

2.2 Получение оперативной помощи

Система руководств весьма подробна, но в некоторых случаях может показаться избыточной. Например, если требуется только уточнить формат команды, или список возможных ее опций, нет необходимости в подробном описании. В этом случае есть смысл прибегнуть к оперативной помощи самих программ.

В большинстве случаев оперативная подсказка по данной конкретной команде может быть получена следующей конструкцией:

```
имя_команды --help
```

или

```
имя_команды -h
```

Очень часто вывод подсказки на экран происходит, если команда, требующая обязательного использования параметров и (или) аргументов, введена без них (или параметры и аргументы введены неправильно). Иногда в этом случае выводится не сама по себе справка, а лишь указывается способ ее получения. Например, ответом на ввод команды `cp` (от copy — копировать) без аргументов будет сообщение такого рода:

```
cp: отсутствуют аргументы
Попробуйте cp --help для получения более подробного описания.
```

Если последовать этому совету, на экран будет выведена справка по использованию команды `cp`:

```
Использование: cp [OPTION]... SOURCE DEST
         или:  cp [OPTION]... SOURCE... DIRECTORY
         или:  cp [OPTION]... --target-directory=DIRECTORY SOURCE...
Копирует SOURCE в DEST, или несколько SOURCE в DIRECTORY.
```

За этим последует расшифровка различных опций (параметров). Аналогично и с командами, которые не требуют обязательного использования опций и аргументов, например, командой `ls`. Чтобы получить по ней справку, требуется так же ввести ее с соответствующим параметром:

```
$ ls --help

Использование: ls [OPTIONS]... [FILE]...
Выдает информацию о FILE (текущий каталог по умолчанию).
```

```
Сортирует в алфавитном порядке если ни один из ключей -cftuSUX --sort
не задан.

-a, --all не скрывать файлы начинающиеся с .
-A, --almost-all не выдавать . и ..
```

и т.д. Напротив, некоторые команды, которые не могут использоваться без опций и (или аргументов) при вводе без них выведут краткую справку о своем использовании. Примером является команда `more`: если после нее не указано имя просматриваемого файла (а именно для этого она и предназначена), ответом будет сообщение:

```
usage: more [-dflpcsu] [+linenum | +/pattern] name1 name2 ...
```

то есть краткая (но часто достаточная для уточнения мелких деталей) справка по использованию команды.

2.3 Экранная документация info

Если Manual Pages — формат интерактивного руководства, общепринятый во всех UNIX-системах, как открытых, так и коммерческих, то система документации `info` разработана в рамках проекта GNU и используется практически только в Linux-системах. По содержанию и назначению документация `info` аналогична man-страницам, но предназначена не только для вывода на экран, но и для подготовки печатных документов.

Для вывода документации info используется команда

```
info имя_программы
```

Например, начальный фрагмент info-документации для команды `ls` будет выглядеть следующим образом:

```
'ls': List directory contents
=============================

   The 'ls' program lists information about files (of any type,
including directories).  Options and file arguments can be intermixed
arbitrarily, as usual.
```

Система `info` имеет довольно сложную на первый взгляд внутреннюю навигацию. Как и в случае man-страниц, команда

```
info info
```

выводит на экран документацию о ней самой, однако очень краткую. Более полную справку по использованию `info` можно получить, запустив из нее встроенную систему помощи. Так, если, находясь на info-странице,нажать комбинацию клавиш `Ctrl`+`H`, на экран будет выведена справка об основных командах в окнах `Info`:

```
Основные команды в окнах Info
******************************

  l          Покинуть эту справку.
  q          Покинуть также Info.
  h          Вызвать обучающее руководство.
```

и т.д. Из нее можно видеть, что нажатие клавиши `l` приведет к возврату на основную info-страницу, а клавиша `h` вызовет подробное руководство, касающееся команды, определенной как аргумент команды `info` (в данном случае — также info), подобно контекстно-зависимым системам помощи во многих Windows-программах. Для перемещения внутри info-страниц служат почти те же клавиши, что и для man-страниц (`Пробел`, `PageUp`, `PageDown` — постраничный просмотр, `Up`и `Down` — построчный), для выхода - клавиша `q` и комбинация `Ctrl`+`C`.

В дистрибутиве **ASPLinux** многие из info-страниц переведены на русский язык. Условия их использования — те же, что и для man-страниц: наличие и корректная локализация.

Вооружившись знанием методов получения информации, можно переходить собственно к изучению приемов использования **ASPLinux** в практической работе.

Глава 3

Командные оболочки

3.1 Вводные замечания о командных оболочках

После загрузки ОС **ASPLinux** и авторизации в системе пользователь, как правило, оказывается в среде командной оболочки, именуемой также командной строкой, интерпретатором команд, командной средой или, по-английски, просто Shell. Это своего рода аналог `COMMAND.COM` из MS DOS. Однако, в отличие от последнего, командная оболочка Linux — это не просто интерпретатор команд, встроенных и внешних, но и интерпретатор мощного языка программирования, позволяющего простыми средствами создавать пользовательские сценарии (скрипты) для решения самых разнообразных задач.

Второе важное отличие командных оболочек Linux от командного интерпретатора MS DOS — их разнообразие. Если в MS DOS `COMMAND.COM` является фактически безальтернативной средой исполнения команд, то в состав любого дистрибутива Linux (и, разумеется, **ASPLinux**) входит большое количество оболочек, среди которых пользователь может выбрать адекватную своим задачам или просто наиболее ему импонирующую.

С одной стороны, командные оболочки Linux различаются по своей функциональности в интерактивном режиме. Выделяется класс простых оболочек, включающих небольшое количество встроенных команд и не поддерживающих широкий круг дополнительных возможностей. Это либо достаточно старые программы, сохраняющие ныне лишь исторический интерес, либо оболочки для специальных применений (например, на спасательных дискетах, где определяющим является минимизация требований к ресурсам). Современные же оболочки поддерживают полный спектр дополнительных возможностей, таких как удобное редактирование командной строки, историю команд, автодополнение и т.д.; число же встроенных команд в современных оболочках исчисляется многими десятками.

С другой стороны, командные оболочки различаются синтаксисом интерпретируемого языка. В этом аспекте также выделяется два их класса, Shell-

совместимые и C-совместимые. Синтаксис языка последних, как нетрудно догадаться из названия, сходен с языком программирования Си, тогда как в Shell-совместимых оболочках он своеобразен.

Кроме специальных командных оболочек, в качестве рабочих сред текстового режима можно использовать и другие программы — интерпретаторы языков программирования (например, tclsh — интерпретатор языка Tcl), файловые менеджеры типа Midnight Commander и даже текстовые редакторы. Однако это целесообразно только в специальных случаях, которые в этом руководстве не рассматриваются.

Командная оболочка — это атрибут учетной записи пользователя. Каждому пользователю может быть назначена определенная командная оболочка. Завершение работы программы, используемой в качестве оболочки, автоматически приводит и к завершению данного пользовательского сеанса. Об этом следует помнить, особенно если в качестве оболочки используется какая-либо альтернативная программа.

В системах Linux (и в дистрибутиве **ASPLinux**) традиционной командной оболочкой по умолчанию является одна из производных классической Shell — `Bourne-Again Shell` (bash). Она принадлежит к категории развитых оболочек, в полной мере реализуя богатство их возможностей.

В принципе все современные оболочки близки по своим возможностям. Поэтому использование той или иной из них — дело вкуса и привычек пользователя. В любой момент можно легко поменять свою рабочую среду: либо редактированием своей учетной записи (об этом подробно говорится в *«Руководстве администратора»*), либо просто запуском новой оболочки внутри старой. Однако ряд системных конфигурационных файлов Linux представляет собой сценарии оболочки (скрипты), и для них обязательным требованием является совместимость по синтаксису со средой Shell. Именно поэтому, а также отдавая дань традиции и функциональности, мы рассматриваем оболочку `bash`.

3.2 Представление о командах оболочки

Как и следует из названия, основное назначение командной оболочки — ввод и исполнения команд. Для ввода служит т.н. командная строка, содержащая приглашение к вводу. По умолчанию в **ASPLinux** она имеет вид

```
[username@localhost username]$
```

при авторизации от имени обычного пользователя, и

```
[root@localhost root]#
```

если запущен сеанс администратора. Группа символов в квадратных скобках — сообщение командной строки, — указывает на имя пользователя (username или root), имя машины (в данном случае — localhost), текущий каталог, которым в момент авторизации является домашний каталог пользователя (username или root). Символы $ или # — собственно приглашение к вводу, — традиционно различаются для обычного пользователя и администратора. Приведенные формы — $ и #, соответственно, — приняты по умолчанию практически во всех оболочках для всех UNIX-систем. Однако пользователь, как будет показано ниже, может менять вид приглашения командной строки.

Далее в командной строке вводятся собственно команды — некие последовательности символов, которые нажатием клавиши Enter отправляются на исполнение. Например, команда

```
ls
```

выведет на экран список файлов, содержащихся в текущем каталоге. Команды могут быть внутренними для данной оболочки и внешними (общесистемными). Последние — это просто любые исполняемые двоичные файлы, расположенные обычно в каталогах /bin, /sbin, /usr/bin и т.п. Собственно к оболочке они не имеют никакого отношения. Примером такой команды является приведенная выше /bin/ls.

Внешних команд, предназначенных для запуска из командной строки, в Linux насчитывается очень много. Представление о их количестве можно получить, дважды нажав в пустой командной строке клавишу табуляции. В ответ будет выведено сообщение, подобное следующему:

```
Display all 2560 possibilities? (y or n)
```

где конкретное число зависит от установки дистрибутива. Если согласиться с предложением, нажав клавишу y, на экран будут выведены все доступные в системе команды.

Следует подчеркнуть, что абсолютно любая программа для Linux может быть запущена из командной строки, хотя и не обязательно в консоли — для запуска программ графического режима потребуется предварительный запуск X Window System и вызов их из командной строки окна терминала. Все прочие способы запуска программ (например, из стартовых меню или панелей интегрированных сред и оконных менеджеров, иконками на их рабочих столах и т.д.) являются дополнительными к этому универсальному способу.

Внутренние команды встроены в саму оболочку, т.е. им не соответствуют никакие исполняемые файлы. Так, бесполезно было бы искать в составе файловой системы файлы, соответствующие командами cd (переход в другой каталог) или pwd (абсолютный путь к текущему каталогу). Полный список внутренних команд bash (а их более 50) можно получить командой help.

Кроме встроенных и внешних команд, в среде командной оболочки могут исполняться т.н. пользовательские сценарии, или скрипты. Подобно внешним командам, это также исполняемые файлы, однако не бинарные, а текстовые. Они составляются в соответствии с синтаксисом языка оболочки и интерпретируются ею при запуске. Подробнее вопрос о сценариях будет рассмотрен ниже.

Большинство команд любого рода исполняется при наличии некоторых параметров, или опций, уточняющих условия их выполнения, и аргументов, в отношении которых команды исполняются. Наиболее часто это выглядит следующим образом:

```
имя_команды --опция1 ... --опция99 аргумент1 ... аргумент99
```

или, в сокращенной форме,

```
имя_команды -abcd аргумент1 ... аргумент99
```

Команды, опции и аргументы обязательно отделяются пробелами. Если имена опций даются в полной форме (например, `--verbose`), при использовании более чем двух опций они также разделяются пробелами, и перед каждой ставится двойной дефис. При сокращенной нотации перед всей группой опций (каждая из которых передается первым символом полного наименования) ставится одинарный дефис. Впрочем, это наиболее типичная, но не обязательная форма. Некоторые команды допускают только полную нотацию опций, например, для получения справки о команде копирования может использоваться только

```
cp --help
```

но не

```
cp -h
```

С другой стороны, существуют команды, не нуждающиеся в знаке дефиса перед опциями, например, команда `tar` для работы архивами. В любом случае, исчерпывающие сведения о возможных опциях и аргументах подавляющего большинства команд Linux можно получить, посмотрев соответствующие страницы руководства.

3.3 Обзор наиболее употребимых команд

Команды оболочки — очень эффективное средство для выполнения многих постоянно использующихся операций. С их помощью осуществляется навига-

ция по файловой системе, поиск файлов, их создание, объединение, копирование, перемещение, удаление и многое другое.

Для вывода сообщений на экран служит встроенная команда echo, имеющаяся и в большинстве других оболочек. Она просто выводит на экран текст, введенный в виде аргумента. Например, команда

```
echo Hello, world!
```

выведет соответствующее сообщение:

```
Hello, world!
```

А в форме

```
echo $SHELL
```

она сообщит название используемой в данном сеансе оболочки; в случае оболочки bash ответом на нее будет

```
/bin/bash
```

Таким образом, команда echo может использоваться для получения информации о переменных окружения, таких, как пути к исполняемым файлам, абсолютный адрес домашнего каталога пользователя, системный редактор по умолчанию и т.д. (см. ниже).

Далее, группа команд предназначеная для просмотра содержимого файловой системы и навигации по ней. Первой цели служит многократно упоминавшаяся команда ls.

Введенная без опций и аргументов, она выводит на экран список файлов текущего каталога, включая подкаталоги. Однако, список этот будет неполон, так как по умолчанию не выводятся имена т.н. скрытых файлов и подкаталогов.

В качестве аргумента команды ls можно указать имя некоторого файла из текущего каталога — ответом будет вывод информации о нем на экран. Вместо имени файла можно указать некий шаблон — результатом будет вывод списка всех файлов, удовлетворяющих этому шаблону. Например, командой

```
ls *.html
```

будет выведен список всех документов HTML текущего каталога.

Если в качестве аргумента команды ls указать путь до некоторого подкаталога, то будет выведен список содержащихся в нем файлов и подкаталогов.

Путь может указываться в абсолютной форме (от корневого каталога) и в относительной (от текущего каталога). Первая форма требует обязательного указания символа / (означающего корневой каталог всей файловой системы):

```
ls /home/userdir/subdir1
```

ответом на что будет список

```
file1  file2 ... file99
```

Относительный путь может задаваться различными путями. Например, если текущим является домашний каталог пользователя (/home/userdir), то команда

```
ls subdir1
```

выдаст список файлов в подкаталоге /home/userdir/subdir1. Каталог, находящийся одним уровнем выше текущего, обозначается символами .. (две точки). И, соответственно, при наличии в домашнем каталоге двух подкаталогов — /home/userdir/subdir1 и /home/userdir/subdir2, содержание второго можно просмотреть, находясь в первом, следующей командой:

```
ls ../subdir2
```

Наконец, для обозначения домашнего каталога данного пользователя приняты обозначения ~/ или $HOME (где $HOME — имя пользователя). Таким образом, командой

```
ls ~
```

или эквивалентной ей командой

```
ls $HOME
```

пользователь может просмотреть содержимое своего домашнего каталога, вне зависимости от текущего каталога (для обозначения которого, к слову сказать, принят символ точки).

Все описанные выше соглашения о путях относятся не только к команде ls, но и к любым другим командам Linux.

Кроме аргументов, команда ls имеет многочисленные опции. Так, опция -l (от list) предписывает выводить полную информацию о содержащихся

в каталоге файлах, включая права доступа к ним, количество ссылок, имя владельца и наименование группы, размер файла, дату и время создания (содержание всех этих атрибутов детально рассмотрено в *«Руководстве администратора»*). Для примера, команда

```
ls -l ~
```

выведет информацию о пользовательском каталоге в полном формате, например, как в ниже приведенном фрагменте:

```
drwxrwxr-x    6 alv      alv          4096 Июн 12 12:23 office52
drwxrwxr-x    6 alv      alv          4096 Июн 10 20:56 other
-rw-rw-r--    1 alv      alv            55 Июн 19 14:43 probe.txt
```

Полностью ознакомиться с опциями команды `ls` можно при помощи команды `man ls`. Заметим только, что опция `-a` (или `--all`) выводит все файлы каталога, в том числе и помеченные как скрытые, а опция `-F` предписывает отличать имена каталогов от имен файлов косой чертой, то есть

```
ls -aF
```

выведет список, подобный следующему фрагменту:

```
.xmms/ soft/ .joerc .xnviewrc .bash_history .kde/ Desktop/
```

где имена вида `.joerc` являются скрытыми файлами, а `soft/` — каталогам. Как нетрудно догадаться, имена вида `.kde/` — суть скрытые каталоги.

Первое, что требуется для навигации по файловой системе — это определение положения текущего каталога в ее структуре, так как от этого зависят все дальнейшие действия. Этой цели служит команда `pwd`. Она не требует ни опций, ни аргументов, выводя в ответ полный абсолютный путь до текущего каталога. Например, сразу после авторизации пользователя это будет:

```
/home/username
```

Вторая из навигационных команд, `cd`, служит для перехода из каталога в каталог. Это внутренняя команда оболочки, не имеющая опций. В качестве ее аргумента выступает путь (абсолютный или относительный) к каталогу назначения, который подчиняется всем правилам, описанным для команды `ls`. Таким образом, команда

```
cd ..
```

обеспечивает переход в каталог верхнего уровня,

```
cd /
```

в корневой каталог и т.д. Конструкции

```
cd $HOME
```

и

```
cd ~
```

предписывают переход в домашний каталог данного пользователя, как и команда cd без указания аргумента.

Важным аспектом навигации по файловой системе является поиск файлов. Для этого предназначена команда find, отличающаяся своеобразным синтаксисом:

```
find [path] -option argument -action
```

Здесь path — это каталог, с которого следует начинать поиск, -option — критерий поиска, -action — действие в отношении найденного файла. В качестве опции могут выступать практически любые атрибуты файла, значения которых указываются в качестве аргумента: имя (-name) с аргументом — именем файла, владелец (-user), группа, к которой приписан файл (-group), тип файла (-type) с аргументами — обычный файл (f), символическая ссылка (l), каталог (d) и т.д. Возможные акции в отношении найденных файлов — вывод имен (с указанием полного пути) на экран в кратком (-print) или полном (-ls) формате, исполнение определенной команды (-exec) и т.д. Например, по команде

```
find /usr -type l -ls
```

будет произведен поиск символических ссылок, начиная с каталога /usr, и результат поиска будет выведен в виде списка файлов с указанием их атрибутов (владельцев, прав доступа, даты создания и т.д.).

При задании имен файлов как аргументов поиска также можно применять шаблоны, однако символ шаблона должен предваряться символом \. Например, команда

```
find ~/ -name \*.html -print
```

предписывает найти все html-документы в домашнем каталоге данного пользователя. Однако, следует заметить, что расширение имени файла (в данном случае — *.html) в Linux никак не связано с его типом и употребляется для удобства или для совместимости с другими ОС.

Следующая группа команд предназначена для управления файлами — их копированием (cp — от copy), перемещением и переименованием (mv — от move), удалением (rm — от remove). Рассмотрим их по порядку.

Команда cp имеет два аргумента — имя исходного файла и имя файла целевого.

То есть конструкция вида

```
cp file1 file2
```

скопирует файл file1 в файл file2 в пределах текущего каталога, то есть в итоге в нем будет два идентичных по содержанию файла, различающихся лишь именами. Напротив, команда

```
cp ~/subdir1/file1 ~/subdir2/
```

скопирует указанный файл в другой каталог с тем же именем; обратите внимание, что имя файла во втором аргументе отсутствует. Хотя никто не запрещает задать его отличным от имени исходного файла, в результате чего в двух каталогах будут присутствовать одинаковые файлы с разными именами — подобно тому, как это имело место при копировании внутри одного каталога.

Особенностью команды cp является то, что если в целевом каталоге имеется файл с именем, совпадающим с именем исходного файла, содержание его будет уничтожено: оно заместится содержанием первого. И потеря эта — безвозвратна. Потому из многочисленных опций команды cp одной из важнейших является опция -i (--interactive), которая вынуждает программу запрашивть подтверждение на исполнение необратимых действий, как-то перезаписывания файла в целевом каталоге.

Обычно во избежание случайного уничтожения файла при копировании (и тому подобных необратимых действиях) опция -i включается как действующая по умолчанию (как — будет рассказано в разделе о настройке оболочки). В дистрибутиве **ASPLinux** такое правило действует по умолчанию для администратора системы (root), который в состоянии уничтожить не только собственные файлы, но и любые другие файлы. Обычные пользователи должны сами позаботиться о сохранности своих данных.

Команда mv (от move — двигать) служит как для переименования, так и для перемещения файлов. В первом случае ее формат

```
mv file1 file2
```

в результате чего в текущем каталоге файл-источник замещается целевым файлом. При перемещении файла, каталоги, содержащие файл-источник и целевой файл, должны различаться:

```
mv ~/subdir1/file1 ~/subdir2/
```

Имя целевого файла можно не указывать. Однако можно использовать эту команду и для переименования файла одновременно с перемещением. При использовании команды mv новый файл не создается, а существующий просто включается в состав другого каталога. В результате он сохраняет все свои атрибуты (дату создания, владельца, права доступа). Командой же cp образуется именно новый файл, отличающийся от файла-источника как минимум датой создания и, возможно, владельцем.

Для всех команд в качестве первого аргумента может быть указано более одного файла, за исключением случаев копирования и переименования файлов в пределах одного каталога.

Команда rm (от remove — удалять) удаляет один или несколько файлов, указанных в качестве ее аргументов.

Кроме указания имен файлов в качестве аргументов могут использоваться их маски (шаблоны). Правила их образования такие: символ ? заменяет любой символ, символ * — последовательность из любого количества символов. Например, команда

```
rm file*
```

удалит не только файлы с именами file1, file2 ... file99, но и файл с именем file.

Опция -i (--interactive) имеет силу и для команд mv и rm. При ее указании запрашивается подтверждение на выполнение действий, к уничтожению каких либо данных. Как и для команды cp, в **ASPLinux** эта опция задействована по умолчанию только для суперпользователя.

Команды cp, rm применимы не только к единичным файлам или их сериям, но и к каталогам. Для этого существует опция -R. Она распространяет действие команд, в качестве аргумента которых указаны имена каталогов, на все входящие в них файлы и вложенные подкаталоги. Так, команды

```
cp -R -iv ~/subdir1/ ~/subdir2/
```

и

```
rm -R -iv ~/subdir1
```

копируют и удаляют, соответственно, каталог subdir1 (со всем его содержимым, включая вложенные подкаталоги). В данных примерах вторая группа опций указывает на необходимость запроса подтверждения действий (-i), во-первых, и вывод сообщений о ходе действий (-v), во-вторых.

Рекурсия, применимая практически ко всем командам Linux, является мощным инструментом для всякого рода файловых операций. Однако она требует осторожного обращения. Так, команда

```
rm -R /
```

данная от лица суперпользователя, уничтожит всю файловую структуру начиная с корневого каталога. [1]

Существуют также команды, специально ориентированные на управление каталогами и неприменимые к отдельным файлам. Так, команда mkdir (от make directory) создает новый каталог, а команда rmdir (от remove dir) — удаляет существующий. В отличие от команды rm с опцией -R, она в состоянии делать это только с пустыми каталогами.

Наконец, серия часто используемых команд предназначена для просмотра содержимого файлов. Первой следует назвать команду cat (от concatenate — объединение, смысл чего станет ясным в следующем разделе). Если в качестве аргумента в ней задать имя существующего файла, она отобразит его содержимое на стандартном устройстве вывода (обычно на экране). Аргументов у команды cat может быть несколько. В этом случае все перечисленные файлы будут выведены один за другим.

Заданная без параметров, команда cat считывает данные со стандартного устройства ввода (обычно клавиатуры). В данном случае это практического смысла не имеет, так как введенные символы исчезнут по завершении работы команды. Однако эта ее особенность будет использована в дальнейшем.

Команда cat только выводит содержимое файла на экран, без возможности перемещения по нему. Конечно, с помощью простой комбинации клавиш Shift+PageUp/PageDown содержимое это может быть просмотрено в буфере экрана, но это обусловлено свойствами консоли Linux, а не оболочки bash или самой команды.

Команда more также предназначена для просмотра содержимого файлов, имена которых (одного или нескольких) обязательны в качестве аргумента. Однако она предоставляет возможность перемещения по файлу, правда, только в одном направлении (от начала к концу) — построчно (с помощью клавиши Enter) или постранично (с помощью клавиши Пробел). Приглашение командной строки в ней возвращается после полного пролистывания файла или после нажатия на клавишу q.

[1] **ВНИМАНИЕ!** Не пытайтесь проверить действие команды на работающей системе!

Команда less, в отличие от more, допускает двунаправленную навигацию по выведенному тексту — вперед и назад, как построчно (клавишами Down и Up), так и постранично (клавишами PageDown и PageUp). Кроме того, для однонаправленного просмотра могут использоваться клавиши Enter и Пробел (как и в more). По достижении конца файла команда less допускает дальнейший его просмотр назад, так как не возвращает приглашения командной строки автоматически — это следует сделать принудительно, нажав клавишу q.

Приведенными примерами список команд, доступных в оболочке bash, не исчерпывается. Некоторые из необходимых и часто используемых команд будут упомянуты в следующих разделах этой главы. Напомним, что более подробные сведения о них можно получить из интерактивной документации — с помощью конструкций man имя_команды или info имя_команды.

3.4 Параллельное выполнение команд

Многозадачность — одно из неотъемлемых свойств ОС Linux, реализуемое в любом режиме ее работы — как в графическом (при помощью оконных интерфейсов разного рода), так и в текстовом.

Один из наиболее эффективных (и эффектных!) способов реализации многозадачности текстового режима — запуск программ в различных виртуальных консолях, переключение между которыми осуществляется с помощью комбинации клавиш Alt+F#, где # — номер соответствующей виртуальной консоли. Однако эта возможность реализуется не средствами командной оболочки, а благодаря свойствам системной консоли Linux. Подробнее о виртуальных консолях будет говориться в *«Руководстве администратора»*.

Оболочка bash допускает последовательное выполнение команд - одна за другой. Для этого серия команд вводится одной строкой, и разделяется символом ;. В этом случае сначала выполняется команда 1, по ее завершении — команда 2 и т.д. Так, в конструкции

```
cd $HOME; ls *.html; cp ./*.html $HOME/web/
```

сначала осуществляется переход в домашний каталог пользователя, затем выводится список всех содержащихся в нем HTML-документов и, наконец, они копируются в подкаталог ~/web/.

Правда, это, в сущности, не многозадачность, так как команды выполняются одна за другой, а лишь технический прием, эквивалентный вводу трех последовательных команд. Однако оболочка bash располагает и собственными средствами доступа к истинной многозадачности. Это запуск программ в фоновом режиме и метод контроля задач.

Для запуска программы в фоновом режиме в командной строке достаточно ввести после всех опций и аргументов команды, знак амперсанда (символ &). Результатом будет возвращение приглашения командной строки сразу после нажатия клавиши Enter, не дожидаясь завершения запущенной команды.

В фоновом режиме имеет смысл запускать долговыполняюшиеся команды и не требующие интерактивного вмешательства пользователя, например, копирование больших массивов данных, архивирование файлов и т.д. В иных случаях можно прибегнуть к методу контроля заданий (jobs control). Он заключается в приостановке выполнения текущей задачи комбинацией клавиш Ctrl+Z, возвращающей приглашение командной строки, в которой запускаются любые команды в любом количестве, каждая из которых, в свою очередь, может быть приостановлена комбинацией клавиш Ctrl+Z.

При необходимости возврата к приостановленной задаче достаточно дать команду fg, в качестве аргумента которой задается ее номер. Узнать этот номер можно, дав предварительно команду jobs, которая выведет список всех приостановленных процессов в следующей форме:

```
[1]   Stopped                 man bash
[2]   Stopped                 vi /tmp/script.sh
[3]-  Stopped                 less cur/aspbooks/install.txt
[4]+  Stopped                 lynx cur/onix/index.html
```

Цифра в квадратных скобках слева и есть номер искомой задачи, следующая колонка — состояние задачи (в данном примере Stopped — приостановленная), и последняя — ее имя. Следовательно, чтобы вернуться к просмотру документации по оболочке bash, следует дать команду fg 1, для перехода к просмотру html-файла в браузере Lynx — прервать просмотр man-страницы и дать команду fg 4, и так далее.

3.5 Комбинирование команд

Команды bash могут использоваться не только сами по себе, но и объединяться в конструкции с помощью двух приемов — перенаправления ввода/вывода и конвейеров команд. Начнем с первого.

Во всех примерах предыдущего раздела команды получали данные со стандартного устройства ввода (клавиатуры), отправляя результаты своей работы на стандартное устройство вывода (экран монитора). Однако и ввод, и вывод команды могут быть перенаправлены. Например, команда может получать данные из уже существующего файла (перенаправление ввода) и (или) выводить результаты в файл (перенаправление вывода). Наиболее часто используется второй вариант. Простейший пример такого перенаправления:

```
ls > listing1
```

При этом список файлов текущего каталога не выводится на экран, а сохраняется в виде нового файла с именем listing1. Символ > означает перенаправление вывода. При повторении этой операции новые данные, перенаправленные на вывод, подменят те, что были сохранены ранее. То есть, в приведенном примере, повторение команды

```
ls > listing1
```

приведет к перезаписи файла listing1. Чтобы избежать этого, вместо простого перенаправления вывода, можно использовать добавление вывода, обозначаемое символом >>. Если в рассматриваемом примере после первого исполнения

```
ls > listing1
```

применить добавление вывода

```
ls >> listing1
```

то новый листинг каталога не перепишет старый файл, а будет добавлен в его конец.

С помощью перенаправления вывода можно создавать новые файлы произвольного содержания. Так, команд cat, заданная без аргумента, просто воспринимает последовательности символов, вводимые с клавиатуры, и воспроизводит их на экране. Если же перенаправить ее вывод, то вводимые с клавиатуры данные можно записать в файл. Делается это следующим образом.

Сначала даем команду

```
cat > newdata
```

и нажимаем Enter. Курсор переходит на новую строку, в результате чего команда готова к приему данных с клавиатуры. Набираем требуемые символы, как в любом текстовом редакторе, используя Enter для создания новых строк:

```
Однажды в студеную зимнюю пору
Я из лесу вышел --- был сильный мороз.
```

и т.д. По завершении набора опять нажимаем Enter (это необходимо для окончания последней строки) и нажимаем комбинацию клавиш Ctrl+D, символизирующую конец файла (аналогично Ctrl+Z в командном интерпретаторе MS DOS) и возвращающую приглашение командной строки. Теперь с помощью команды

```
ls newdata
```

легко убедиться, что новый файл был успешно создан, а командой

```
cat newdata
```

(а также less или more) — проверить его содержимое.

При вводе в команде cat возможно только самое элементарное редактирование, а именно: уничтожение всех символов строки вплоть до требующего изменения, а затем набор их заново. С переходом на новую строку предыдущая становится недоступной для изменения. Поэтому таким способом неудобно создавать сколько-нибудь объемные документы, но использование его для создания коротких записей (например, простых сценариев оболочки) вполне целесообразно.

Благодаря перенаправлению вывода раскрывается вся мощь команды cat (а заодно становится понятным ее название). С ее помощью можно добавить данные к существующему файлу с клавиатуры

```
cat >> olddata
```

(тем же способом, что и создать новый файл) или из существующего файла

```
cat newdata >> olddata
```

Можно также слить вместе произвольное количество файлов:

```
cat file1 file2 ... file99 > allfiles
```

Кроме перенаправления вывода, имеется возможность и перенаправления ввода (символ <). Это получение командой данных не с клавиатуры, а из существующего файла. Так, если команда

```
cat > salute
```

с введенным

```
Hello, world!
```

Ctrl+D

создает файл salute, то обратная ей команда

```
cat < salute
```

выведет его содержимое на экран в виде

```
Hello, world!
```

Легко видеть, что это эквивалентно просто команде `cat` с именем соответствующего файла в качества аргумента. И потому само по себе перенаправление ввода используется редко. Однако оно может применяться в более сложных конструкциях, объединяющих перенаправление как ввода, так и вывода. Так, ранее созданный перенаправлением вывода файл `listing1` содержит список файлов, отсортированный командой `ls` по умолчанию, то есть в порядке возрастания ASCII-кода первого символа (это не совсем то же самое, что в алфавитном порядке). Это положение можно изменить, применив команду `sort` (предназначенную для сортировки записей) с опцией `-r` (от `reverse`, то есть в обратном порядке) и операцию перенаправления ввода:

```
sort -r < listing1
```

Она выведет на экран отсортированное в обратном порядке содержимое файла, из которого берутся данные. Если же перенаправить ее вывод

```
sort -r < listing1 > listing2
```

результатом будет создание нового файла с тем же списком, что и в исходном, но иным порядком сортировки.

Второй прием для объединения командных конструкций — конвейеры команд. Иногда они именуются в литературе каналами, но название «конвейер» лучше отражает существо дела. В этом случае вывод одной программы направляется не в файл, а на стандартный ввод другой программы. Простой пример такой операции — просмотр списка файлов:

```
ls -l | less
```

Перенаправление вывода команды `ls`, то есть списка файлов, который при использовании полного формата записи (опция `-l`) может занимать многие экраны, позволяет просматривать результат с помощью команды `less` постранично или построчно в обоих направлениях.

Конвейеризация команд может быть сколь угодно длинной. Так, конструкция

```
ls -lt | sort -r | lpr
```

обеспечит не только сортировку списка файлов каталога в полном формате, выведенного командой ls по времени изменения (опция -t), в обратном относительно исходного порядке (опция -r команды sort), но и выведет его на печатающее устройство командой lpr.

В составе командных конструкций полезной может оказаться команда tee. Она обеспечивает получение данных по конвейеру от предыдущей команды и одновременный вывод их на экран для просмотра и в файл для записи:

```
cat file1 ... file99 | tee allfiles
```

Приведенные примеры использования перенаправления ввода/вывода могут показаться искусственными и не имеющими практического значения. Это не так: в ряде случаев они позволяют простым и эффективным способом решать реальные пользовательские задачи.

Приведем один из наиболее показательных примеров. Имеется: серия текстовых файлов (например, глав и фрагментов настоящего руководства - ch1, ch2 и т.д.), рассеянных по различным каталогам (dir1, dir2 и т.д.) файловой системы. Требуется собрать из них в определенном порядке (с главы первой по последнюю) единый документ (book) — то руководство, которое вы читаете, записать его в новый файл, который должен быть помещен в собственный каталог. Предлагается подсчитать, сколько действий потребуется для решения этой задачи (согласитесь, более чем жизненной) в любом текстовом редакторе или текстовом процессоре. Средствами же оболочки bash это выполняется одной командой:

```
cat /dir1/ch1 /dir2/ch2 ... /dir10/ch10 > /pathbook/book
```

Включив командный конвейер, можно одновременно просмотреть на экране результат и распечатать его на принтере:

```
cat /dir1/ch1 /dir2/ch2 ... /dir10/ch10 | tee /pathbook/book | lpr
```

При этом не следует пугаться, что придется вручную (да еще и многократно) набирать с клавиатуры длинные и причудливые пути к файлам: в bash предусмотрены средства автоматизации этого процесса, к рассмотрению которых мы и переходим в следующем разделе.

3.6 Автоматизация интерактивной работы

Как можно было заметить в примерах предыдущего раздела, для выполнения большинства действий в командной строке оболочки требуется ввод разнообразных команд, часто с множеством опций и аргументов, требующих подчас

указания длинных путей к исполняемым файлам и файлам данных. Однако оболочка bash располагает тремя базовыми средствами автоматизации, делающими интерактивную работу в командной строке простой и легкой. И к тому же избавляющей от необходимости запоминать целиком команды и полные пути к файлам. Это автодополнение команд и путей, история команд и редактирование командной строки.

Чтобы воспользоваться свойством автодополнения, достаточно набрать один или несколько первых символов команды и нажать клавишу табуляции Tab. Если введенных символов хватает для однозначного определения команды, недостающие будут добавлены автоматически. Если нет — последует звуковой сигнал, после которого вторичное нажатие клавиши Tab выведет на экран список совпадающих с шаблоном команд, из которых можно выбрать требуемую в данном случае.

Иногда количество команд, отвечающих заданному шаблону, оказывается довольно большим, и тогда задается вопрос, выводить ли все команды из такого-то их количества, или нет. Просматривать столь длинные списки команд на экране неудобно и потому обычно лучше ввести еще один или несколько уточняющих символов.

Как уже говорилось выше, нажатие клавиши табуляции при пустой командной строке выведет полное количество команд, доступных системе в соответствии с переменной окружения $PATH, о которой будет сказано в разделе про настройку оболочки.

Автодополнение применяется не только по отношению к командам, но и к путям файлов, используемым в качестве их аргументов, как абсолютным, так и относительным. Действие его аналогично: набирается некоторое количество начальных символов пути в абсолютной (начиная с символа /) или относительной (без него) форме и нажимается клавиша Tab. Если их окажется недостаточно, следует либо ввести дополнительные символы для уточнения, либо, после повторного нажатия клавиши Tab, выбрать подходящий вариант из списка.

К сожалению, свойство автодополнения не распространяется на опции команд, например, напрасно было бы пытаться, набрав

```
ls --he
```

использовать табулятор для получения справки по команде ls. Не удается использовать автодополнение также для имен команд, используемых как аргументы команды man.

Второй способ автоматизации ввода обеспечивается свойством bash хранить историю введенных команд в количестве, определяемом ее настройками. В дистрибутиве **ASPLinux** по умолчанию оно равно 1000.

Для просмотра истории команд существует одноименная внутренняя команда оболочки — `history`. Она выводит на экран нумерованный список всех вводившихся команд примерно следующего вида:

```
997 startx
998 less admin.txt
999 less user.txt
1000 find / -name locate -print &
1001 logout
1002 joe hist.txt
1003 history
```

Любая из них может быть исполнена повторно, для чего требуется, определив ее номер, ввести его в командной строке, предварив символом `!` без пробела между ним и номером, например,

```
!1000
```

повторит поиск файла `locate`, определенного как аргумент команды `find`. Вместо номера после символа `!` можно указать один или несколько символов из начала имени требуемой команды. В приведенном примере команда `!j` вызовет текстовый редактор `joe` для редактирования файла `hist.txt`, а команда `!s` запустит сеанс X Window System, то есть наиболее поздние из совпадающих с шаблоном.

Если введенные символы совпадают более чем с одной командой из истории, будет исполнена последняя по списку. Скажем, при введении команды `!l` из приведенного выше списка будет выполнена команда `logout`, если же задать в командной строке `!le`, повторится просмотр командой `less` файла `user.txt`, но не `admin.txt`.

Хотя количество команд, хранимых в истории, составляет, как уже говорилось, 1000, для непосредственного просмотра их через пролистывание экранного буфера (посредством комбинации клавиш `Shift`+`PageUp`) доступна лишь часть их, зависящая от размера буфера, что определяется не настройками оболочки `bash`, а свойствами консоли Linux. Однако есть и другие способы доступа к истории команд.

Наиболее простой, хотя и наименее эффективный из них, — это просмотр истории команд с помощью клавиш управления курсором `Up` и `Down` (вызывающих, соответственно, предыдущую и последующую команды). Эквивалентные им комбинации клавиш — `Ctrl`+`P` (от `previous`) и `Ctrl`+`N` (от `next`) соответственно. Кроме того, комбинацией клавиш `Alt`+`<` можно перейти к первой команде из сохраненных в истории, а `Alt`+`>` — к последней. Однако при обширной истории команд любая из этих процедур может быть весьма длительной.

Наконец, комбинация Ctrl+R обеспечивает прогрессивный поиск нужной команды в истории команд по одному или нескольким составляющим ее символов. Выполняется поиск следующим образом: после нажатия указанной клавишной комбинации появляется предложение ввести часть искомой строки:

```
(reverse-i-search)':
```

Последовательный ввод символов выведет последнюю из команд, их содержащую. Введенные символы не обязательно должны входить в имя команды, а могут быть составляющими ее опций или аргументов (имени файла или пути к нему, например). Следующее нажатие Ctrl+R переведет курсор к следующему элементу истории, содержащему введенные символы.

Например, если команда `history` вывела на экран список следующего вида:

```
1005 cd cur/aspbooks/
1006 less admin.txt
1007 lynx index.htm
1008 logout
1009 less user.txt
1010 ls -l
1011 history
```

и если после нажатия Ctrl+R для поиска задать символ `l`

```
(reverse-i-search)': l
```

первой будет найдена команда

```
(reverse-i-search)': ls -l
```

с фиксацией курсора на опции `l`. При повторном нажатии Ctrl+R курсор переместится в начало команды `ls`, при следующем будет выведена команда

```
(reverse-i-search)': less user.txt
```

далее

```
(reverse-i-search)': logout
(reverse-i-search)': lynx index.htm
```

и, наконец,

```
(reverse-i-search)': less admin.txt
```

Нажатие клавиши Enter в любой из этих моментов запускает найденную команду на исполнение. Поиск обрывается также и при нажатии любых иных клавиш или их комбинаций, либо поиск продолжается дальнейшим нажатием Ctrl+R.

Все сказанное выше относилось к тем случаям, когда требовалось повторить ранее использовавшуюся команду в неизменном виде. Более частой, однако, является ситуация, когда необходимо повторить команду, изменив некоторые из ее опций и (или) аргументов. На этот случай в bash предусмотрено свойство редактирования командной строки.

Разумеется, в любой выбранной из истории команде можно просто удалить (клавишей Backspace) все символы вплоть до того, который требует изменения, заменить его и ввести все заново. Однако есть и более целесообразные с точки зрения экономии усилий способы.

Навигация внутри введенной команды, ее опций и аргументов возможна с помощью клавиш управления курсором — клавиша Left перемещает его на один символ влево, клавиша Right — на один символ вправо, Home — в начало строки, End — в ее конец. Для редактирования символов, требующих изменения, можно использовать клавиши Del (удаление символа в позиции курсора) и Backspace (перед ней).

Однако есть возможность обойтись только алфавитно-цифровой частью клавиатуры в сочетании с управляющими клавишами Ctrl и Alt , что дает и более широкие возможности для навигации и редактирования. Например, комбинации клавиш (заметьте, что под точкой отсчета понимается текущая позиция курсора в строке):

Ctrl+B

- (от back) перемещает курсор на один символ назад (аналогично клавише Left),

Ctrl+F

- (от forward) — на один символ вперед (подобно клавише Right),

Alt+B

- — назад к ближайшему пробелу, разделяющему опции и аргументы команды (или ближайшему символу в имени файла-аргумента, не являющемуся буквой или цифрой),

Alt+F

- — вперед к ближайшему пробелу (или не буквенно-цифровому символу),

Ctrl+A

- переводит курсор в начало строки,

Ctrl+E

- — в ее конец.

Сходные комбинации используются и для редактирования командной строки.

Сочетание Ctrl+D удаляет символ в позиции курсора, Ctrl+H — перед ней, Alt+D — часть строки до ближайшего справа пробела или точки, Ctrl+W — до ближайшего пробела слева. С помощью Ctrl+K удаляется часть строки справа от позиции курсора. Alt+\ — аналогично Ctrl+H, но удаляются только пробелы.

Комбинацией клавиш Ctrl+T осуществляется смена положения соседних символов (символ в позиции курсора перемещается налево, предыдущий занимает его место), комбинацией Alt+T — смена положения последовательности символов («слов»), разделенных пробелами или точками (по аналогичным правилам).

В разделах `Commands for Moving`, `Commands for Changing Text` и `Killing and Yanking` руководства по `bash` приведено еще множество команд для навигации по командной строке и редактирования ее. Предусмотрена даже функция отмены изменений (Ctrl+-), правда, только одноуровневая.

В целом команды навигации и редактирования в оболочке `bash` подобны аналогичным командам таких текстовых ред:w акторов, как `emacs` и `joe`. Однако это положение по умолчанию можно изменить, и с помощью соответствующих настроек сделать их идентичными командам редактора `vi`.

Интересна возможность редактирования команд, найденных в истории с помощью функции `reverse-i-search` (по комбинации Ctrl+R). В ней можно изменить любые символы, пользуясь изложенными выше приемами. Однако, при этом редактируется не просто данная команда для последующего выполнения, а именно команда в истории. Если в приведенном выше примере в строке 1010

```
(reverse-i-search)': ls -l
```

заменить опцию `-l` на, скажем, опцию `-a`, то при следующем просмотре командой `history` мы увидим за номером 1010 уже измененную команду:

```
1009 less user.txt
1010 ls -a
1011 history
```

Если же удалить команду целиком, в истории команд под этим номером останется пустое поле.

3.7 Работа с мышью в командной оболочке

Может показаться, что для работы в командной оболочке `bash` хватает автозаполнения и других полезных функций, уже описанных ранее. Это не совсем так. Иногда для работы с большими объемами текста гораздо удобнее использовать знакомую нам по процессорам слов типа WYSIWYG мышь. Она может быть с успехом применена для автоматизации некоторых рутинных действий в `bash`.

Правда, обеспечивается это не свойствами оболочки и даже не свойствами консоли Linux, а программой управления мышью `gpm`, запускаемой по умолчанию в качестве одного из стартовых сервисов, подобно резидентному драйверу мыши в MS DOS.

Следует заметить, что в текстовом режиме Linux мышь является не столько указательным устройством (хотя в некоторых программах, например **Midnight Commander**, может выступать и в этом качестве), сколько средством для выделения экранных блоков, копирования их в буфер и вставки в нужные места.

Копирование блока в буфер происходит автоматически, при выделении его мышью. Подчеркнем, что копируется именно блок экрана, а не фрагмент текста. Так, если при выделении строки часть ее уходит за пределы экрана, то в буфер попадет только видимая на мониторе ее часть.

Вставка текста осуществляется щелчком средней кнопки мыши (для двухкнопочных мышей она обычно эмулируется одновременным нажатием двух) и ведет к появлению выделенного фрагмента, начиная с позиции курсора так, как если бы он набран с клавиатуры. Курсор в требуемую позицию перемещается стрелками управления или описанными выше клавишными комбинациями, но ни в коем случае не мышью — она на поведение курсора в консоли Linux влияния не оказывает (за исключением некоторых программ, в которых мышь поддерживается как указательное устройство).

Особенности поведения мыши в консоли Linux позволяют использовать ее для автоматизации работы в командной строке. Любая ранее введенная команда, видимая в данный момент на экране, может быть выделена мышью, вставлена в командную строку, при необходимости отредактирована и нажатием клавиши `Enter` запущена на исполнение.

Выделению и вставке поддаются команды из списка их истории, а также ранее исполнявшиеся команды, ставшие видимыми благодаря пролистыванию экранного буфера клавишами `PageUp`/`PageDown`.

Обратите внимание, что копировать информацию можно между виртуальными терминалами — например, выделив текст на одном терминале, затем, переключившись, вставить на другом.

3.8 Понятие о сценариях оболочки

Сценарии оболочки, или скрипты, — самый эффективный способ автоматизации работы в bash. Скрипты — обычные файлы в текстовом (ASCII) формате, содержание которых составляет одна или несколько команд с их опциями и аргументами, при необходимости соединенные условными операторами. Скрипты являются прародителями bat-файлов в MS DOS, но позволяют реализовать значительно более широкий круг возможностей.

Сценарии оболочки очень широко применяются в Linux. Ими являются большинство конфигурационных файлов, как общесистемных, так и пользовательских, и многие прикладные программы. Кроме того, любой пользователь может создавать собственные сценарии в неограниченном количестве.

В качестве сценария можно сохранить любую часто используемую команду с ее опциями и аргументами, командную конструкцию, объединенную перенаправлением и (или) конвейеризацией, просто серию последовательно выполняемых команд.

Для этого достаточно сделать следующее:

- дать команду на создание файла, например

  ```
  cat > scr1
  ```

  и нажать Enter;
- ввести строку

  ```
  #!/bin/sh
  ```

- перейти на новую строку и любым способом — набором с клавиатуры или копированием мышью, — ввести требуемую команду, серию последовательных команд (в одну или несколько строк) или командную конструкцию

  ```
  command1 -options arguments
  command2 -options arguments
  ```

- перейти на новую строку (нажатием Enter);
- завершить файл (нажатием Ctrl+D);
- разрешить исполнение файла командой chmod.

Кратко прокомментируем эти действия. Первая строка идентифицирует сценарий оболочки. Символ # в оболочке bash (и большинстве других случаев) служит в принципе для обозначения комментария (аналогично комментарию rem в bat-файлах MS DOS). То есть символы, следующие в одной строке за ним, не рассматриваются как команды и не принимаются к исполнению, а служат исключительно внутренним целям, поясняя в необходимых случаях действия командных директив. Если комментарий занимает более чем одну строку, каждая из них должна начинаться с символа #.

Однако первая строка #!/bin/sh имеет иной смысл. Она указывает, что данный сценарий исполняется именно в Shell-совместимой оболочке и, если пользователь использует по умолчанию оболочку иного типа, предписывает запуск внутри нее оболочки sh (почему именно sh, а не, например, bash, будет сказано в одном из следующих разделов).

Ввод собственно команд понятен без комментариев, а вот на последнее действие следует обратить особое внимание. Все файлы в Linux имеют права доступа, разделяемые на права действия и права принадлежности (подробно об этом говорится в *«Руководстве по администрированию»*). Права действия включают в себя право на чтение файла (read, или r), его изменение (write, или w) и исполнение (execute, или x). Права эти могут принадлежать (права принадлежности) владельцам файла (owner, или в других случаях, user, u), членам одной из групп пользователей, в которые входит владелец (group, или g) и всем прочим (other, или o), а также им всем вместе (all, или a).

Пользовательский сценарий, который может быть запущен как команда, отличается от просто текстового файла только правом на исполнение.

Команда chmod (от change mode) и служит для изменения прав доступа к файлам.

Чтобы сделать созданный сценарий доступным для исполнения его создателем (а по умолчанию он является обычно и хозяином файла, owner или user, автоматически имеющим права чтения и записи в отношении него), команда эта дается виде

```
chmod u+x scr1
```

Аргумент команды — это имя нашего скрипта, смысл опций следующий:

- первая опция указывает на принадлежность того, для кого изменяются права доступа; в данном случае это user (u) — владелец файла;
- следующая опция — это характер изменения прав; нетрудно догадаться, что + означает присвоить право, - отнять его; в данном случае некое право в отношении файла присваивается;
- наконец, последняя опция — это именно то самое изменяемое командой право доступа, в данном случае — право на исполнение (x).

Одной командой chmod можно присвоить сразу несколько прав действия и принадлежности. Так, команда

```
chmod ugo+rx scr1
```

устанавливает право не только исполнять (x), но и читать (r) исходный текст созданного сценария не только его владельцу (u, который право на чтение имеет по умолчанию), но и членам его группы (g, для которых право чтения обычно также устанавливается по умолчанию), но и другим пользователям (o).

Можно также одной командой присвоить все возможные права действия и принадлежности в отношении сценария:

```
chmod a+rwx scr1
```

где a (all) означает всех пользователей по принадлежности, r, w и x — права на чтение, запись и исполнение, соответственно (порядок опций, соответствующих отдельным правам как принадлежности, так и действия, может быть любым, но обычно приняты ugo и rwx).

Подчеркнем, что опции a и o в приведенных примерах не эквивалентны: a есть совокупность прав принадлежности владельцу, группе и прочим, под которыми подразумеваются все пользователи, не являющиеся владельцами файла и не входящие в группу пользователей, к которой приписан файл. И потому командой

```
chmod g-rwx scr1
```

можно отнять все права у группы пользователей, сохранив их для себя и всех, в эту группу не входящих. Впрочем, и об этом подробно рассказывается в *«Руководстве по администрированию.»*

Установкой соответствующих прав доступа создание сценария завершается.

Остается только поместить его в подходящий каталог из числа определенных переменной окружения $PATH, например, в $HOME/bin. И теперь сценарий можно запустить, набрав в командной строке короткую и мнемонически прозрачную последовательность символов вместо сложной последовательности команд, их опций и аргументов.

Разумеется, внутри сценария могут использоваться любые командные конструкции с перенаправлением ввода/вывода и конвейеризацией, а также условные операторы типа if ... then ... fi, if ... then ... else ... fi. Однако программирование сценариев оболочки — отдельная тема, которой посвящено большое количество источников информации (см. заключение).

3.9 Настройка bash

Настройка оболочки bash также призвана автоматизировать и упростить интерактивную работу в ней. Так, можно постоянно вводить вручную путь к каталогу, содержащему пользовательские сценарии, а можно, настроив должным образом переменные окружения, вызывать их, находясь в любом каталоге файловой системы. Или, как говорилось выше, вызывать команды типа rm, mv и т.д. с параметром, запрашивающим подтверждение для выполнения потенциально опасных данных, а можно путем определения псевдонимов команд (aliases) предписать вывод такого подтверждения по умолчанию.

Поведение оболочки bash по умолчанию определяется двумя общесистемными конфигурационными файлами — /etc/bashrc и /etc/profile. Они создаются при установке **ASPLinux** и содержат некий первоначальный набор настроек, подходящий для большинства пользовательских задач.

Кроме того, при создании учетной записи пользователя и назначения ему оболочки bash в качестве рабочей среды в его пользовательском каталоге создаются два файла аналогичного назначения — $HOME/.bashrc и $HOME/.bash_profile. Они запускаются сразу после общесистемных, файлами в сеансе конкретного пользователя. Именно их редактированием достигаются индивидуальные настройки.

Иногда пользовательские конфигурационные файлы в домашнем каталоге могут отсутствовать, если, например, при создании учетной записи пользователя на стадии установки **ASPLinux** для него была определена другая оболочка, позднее вручную замененная на bash. В этом случае их следует создать либо вручную, либо копированием из каталога /etc:

```
cp /etc/bashrc $HOME/.bashrc
```

и

```
cp /etc/profile $HOME/.bash_profile
```

либо просто командой cat:

```
cat > .bashrc
```

и

```
cat > .bash_profile
```

В последнем случае их можно оставить пустыми, внося при необходимости только те опции, которые потребуется сделать отличными от общесистемных.

Следует обратить внимание на то, что пользовательские конфигурационные файлы начинаются с символа . (точка). Это общее правило, касающееся не только файлов конфигурации bash, но и почти любых программ. Точку в данном случае следует отличать от обозначения текущего каталога (. или ./) — она означает, что файл с именем такой формы является скрытым (аналог атрибута hidden в MS DOS). Он не будет, в частности, отображаться командой ls по умолчанию.

Чтобы увидеть скрытые файлы в списке текущего каталога, команду эту следует дать в форме

```
ls -a
```

Назначение файлов .bashrc и .bash_profile различно. Первый считывается один раз — при авторизации пользователя и первом запуске оболочки в сеансе. Файл .bash_profile перечитывается всякий раз, как запускается новый экземпляр программы bash, что происходит при исполнении некоторых программ или, например, при запуске окна терминала в X Window System. И потому не следует удивляться, что поведение оболочки bash может быть различным в разных случаях. Хотя ничто не запрещает сделать настройки обоих конфигурационных файлов идентичными.

В домашнем каталоге администратора (/root) также имеются индивидуальные файлы /root/.bashrc и /root/.bash_profile, настройки которых, как правило, отличаются и от общесистемных, и от прочих пользовательских. И потому если авторизоваться в системе как root, поведение оболочки bash будет иным, чем в сеансе обычного пользователя. Однако если последний получает временные привилегии администратора командой su, поведение bash по прежнему определяется его пользовательскими конфигурационными файлами, а не настройками суперпользователя.

В результате пользователь, получив права администратора командой su, иногда не в состоянии запустить некоторые программы, предназначенные для общесистемного конфигурирования. Это может вызвать недоумение, хотя объясняется очень просто: путь к файлам этой группы не определен в переменной окружения конфигурационного файла данного пользователя. И преодолевается еще проще — запуском команды с указанием полного пути (обычно команды такого рода собраны в каталоге /sbin), переопределением переменной окружения в данном сеансе или исправлением своего конфигурационного файла.

Впрочем, при необходимости доступа к окружению администратора в полном объеме, для получения его прав команду su следует использовать в форме

```
su -l
```

(или просто su -). В этом случае происходит перечитывание файла начальной

конфигурации командной оболочки администратора.

Уяснив назначение конфигурационных файлов, можно приступить к их редактированию.

Для этого потребуется любой текстовый редактор, например, описанные в этом руководстве консольные редакторы `vi` или `joe`, или редактор для среды *KDE* — `kwrite`. Обратим только внимание, что если выбранный редактор по умолчанию поддерживает перенос строк по достижению границы экрана или окна, эту опцию следует обязательно отключить: в результате принудительного разрыва строк файл может стать неработоспособным. Об этом следует всегда помнить при редактировании любых файлов конфигурации и скриптов.

Как правило, у пользователя возникает потребность в изменении двух групп настроек — переменных окружения и псевдонимов команд.

Начнем с переменных окружения. Через них определяются такие свойства оболочки, как тип терминала (переменная `$TERM`), домашний каталог пользователя (переменная `$HOME`) и многие другие. В Shell-совместимых оболочках (в том числе и в `bash`) любая переменная задается в одном из конфигурационных файлов `bash` таким образом:

```
ИМЯ_ПЕРЕМЕННОЙ=значение
```

Например, для смены типа используемого терминала на `vt100` следует добавить

```
TERM=vt100
```

Большинство переменных является внутренними (называемыми переменными оболочки), то есть доступными только для внутренних ее команд. Полный их список, который можно просмотреть командой `set`, включает многие десятки. Они используются главным образом при создании сценариев.

На поведение оболочки при интерактивной работе большее влияние оказывают переменные окружения (`shell environment`). Чтобы превратить переменную оболочки в переменную окружения, ее следует экспортировать, то есть сделать доступной для внешних команд. Чтобы проделать это с переменной `$TERM` из приведенного примера, достаточно присоединить к ее определению выражение

```
export TERM=vt100
```

или добавить его отдельной строкой:

```
TERM=vt100
export TERM
```

Обычно переменные окружения определяются в файле $HOME/.bash_profile.

Просмотреть текущие их значения можно командой

```
echo $ИМЯ_ПЕРЕМЕННОЙ
```

Одной из важнейших среди переменных окружения является неоднократно упоминавшаяся ранее переменная $PATH. Настало время остановиться на ней подробнее.

Переменная $PATH задает список каталогов, просматриваемых интерпретатором оболочки при поиске команд, то есть исполняемых бинарных файлов и сценариев. Текущее значение $PATH можно определить командой:

```
echo $PATH
```

ответом на что будет сообщение вида

```
/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/usr:/home/username/bin
```

Это стандартный набор каталогов (значения, разделенные двоеточием) для исполняемых файлов, определяемый общесистемными настройками bash, за исключением последнего компонента, который определяется в файле $HOME/.bash_profile следующим образом:

```
PATH=$PATH:$HOME/bin
```

то есть к общесистемному значению переменной $PATH для любого пользователя присоединяется его персональный подкаталог bin в домашнем каталоге ($HOME/bin), что для конкретного пользователя username принимает форму /home/username/bin.

Если по тем или иным причинам файл $HOME/.bash_profile не был создан одновременно с учетной записью, строку PATH=$PATH:$HOME/bin (или полное перечисление каталогов, как приведено выше) нужно внести в него вручную.

Далее, следует помнить, что командные интерпретаторы оболочек Linux, в том числе и bash, в отличие от COMMAND.COM для MS DOS, не производят поиска исполняемых файлов в текущем каталоге сами по себе, что часто

обескураживает пользователя. Чтобы они делали это, текущий каталог должен быть явно (в форме ./) указан в переменной окружения $PATH. Однако, делать это не безопасно, так как может быть использовано для подмены системных утилит при взломе системы.

Наконец, в перечне каталогов $PATH отсутствуют каталоги для команд административного назначения (/sbin и /usr/sbin). Если пользователю часто приходится обращаться к таким командам после получения временных прав суперпользователя, следует, как уже говорилось, пользоваться командой

```
su -
```

что предпочтительней прямой авторизации в качестве root. Можно также внести каталоги, содержащие административные команды, в $PATH, причем ранее всех прочих. В результате соответствующая строка в $HOME/.bash_profile примет вид

```
PATH=/sbin:/usr/sbin:$PATH:$HOME/bin
```

а ответом на команду

```
echo $PATH
```

будет вывод на экран следующего перечня:

```
/sbin:/usr/sbin:/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/alv/bin
```

Порядок перечисления каталогов в переменной $PATH имеет значение, так как при вводе любой команды интерпретатор bash ищет его слева направо, то есть, в приведенном примере, сначала в /sbin, потом в /usr/sbin и т.д., и лишь в последнюю очередь — в $HOME/bin. Поэтому если пользователь создал собственную команду, одноименную какой-либо системной из каталога /sbin и хочет в первую очередь использовать именно ее, следует позаботиться, чтобы в перечне каталогов $HOME/bin стояло раньше, чем /sbin.

Переменная $PATH, как и любая другая, заданная в форме

```
PATH=...
```

остается пока внутренней переменной оболочки. Для превращения в переменную окружения ее нужно экспортировать. Если файл $HOME/.bash_profile был создан одновременно с учетной записью, это было сделано автоматически. Если нет — после строки PATH=... следует добавить

```
export PATH
```

Через переменные окружения можно настроить вид приглашения командной строки, что имеет не только декоративную, но и функциональную сторону: приглашение призвано помочь пользователю ориентироваться в его положении в структуре каталогов, отличать пользователей друг от друга и от администратора и т.д.

Так, принятая в bash по умолчанию форма

```
[username@localhost username]$
```

показывает имя пользователя, имя машины и имя текущего каталога, а также позволяет с первого взгляда отличить обычного пользователя ($) от администратора (#). Однако ее можно сделать еще более информативной, для чего используется переменная окружения $PS1. Так, если определить ее следующим образом:

```
PS1='\u:\w=>>'
```

в приглашении будет выводится имя пользователя (u) и полный путь до текущего каталога, начиная от домашнего (w, символ \ в обоих случаях указывает, что следующий за ним символ — не просто алфавитный, а имеет специальное значение), а само приглашение приобретет вид

```
alv:~/cur/src=>>
```

Кроме того, в приглашение можно ввести любую последовательность обычных алфавитных символов, например, сообщение вида

```
alv:Введите команду~=>>
```

что достигается строкой

```
PS1='\u:Введите команду\w=>>'
```

Разумеется, переменную $PS1 после настройки нужно экспортировать

```
export PS1
```

Кроме этого, пользователь может определить для себя и другие переменные окружения, такие, как системный редактор по умолчанию

```
EDITOR=joe export EDITOR
```

или программу постраничного просмотра

```
PAGER=more export PAGER
```

Не менее полезным, чем определение переменных окружения, является задание псевдонимов (aliases) для некоторых команд. Это обычно делается в файле $HOME/.bashrc следующим образом:

```
alias имя_псевдонима='имя_команды [-опции]'
```

Имя псевдонима может совпадать с именем команды, опции которой задают условия ее выполнения. Ранее неоднократно говорилось, что во избежание непреднамеренной потери данных команды типа rm, cp, mv лучше задавать с опцией -i, запрашивающей подтверждения потенциально опасных действий (например, при совпадении имен файлов и т.д.). Псевдонимы позволяют автоматизировать этот процесс, сделав такой запрос принудительным.

Для этого в файл $HOME/.bashrc вносятся строки:

```
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

то есть команды rm, cp, mv определяются как псевдонимы самих себя, но с обязательной опцией -i, в результате чего предупреждающие сообщения будут выводиться всегда.

При уничтожении большого количества файлов и уверенности в правильности своих действий псевдоним для rm (или любой другой команды) можно временно отключить командой:

```
unalias rm
```

а по завершении процедуры — восстановить:

```
alias rm='rm -i'
```

Последняя команда может использоваться при интерактивной работе для временного определения псевдонимов.

Можно создавать имена псевдонимов, отличные от имен подменяемых команд. Так, команда ls по умолчанию не выводит имена скрытых файлов (вида .имя_файла), и в большинстве случаев это удобно. Однако при всякого рода настройках видеть скрытые файлы необходимо. И на этот случай можно определить псевдоним типа

```
alias la='ls -a'
```

и внести его в $HOME/.bashrc или использовать интерактивно.

Описанными примерами далеко не исчерпываются возможности настройки bash.

Полную информацию по этому вопросу можно получить из экранной документации man bash.

3.10 Прочие оболочки

Оболочка bash — не единственная используемая в Linux. Исторически первой среди них была собственно оболочка Борна — sh (Bourne Shell), родившаяся вместе с первыми UNIX-системами. Она была реализована для всех UNIX и UNIX-подобных систем и принята в них в качестве стандартной. Многие общесистемные сценарии требуют для своей работы оболочки sh. Именно поэтому в первой строке большинства сценариев приводится строка #!/bin/sh.

Основные отличия sh от bash можно охарактеризовать частицей НЕ: она не поддерживает автодополнения команд и путей при вводе, истории команд, редактирования командной строки, контроля заданий и многих других возможностей, ставших стандартными для современных оболочек. Кроме того, она не является свободно распространяемой.

Все это служит причиной того, что в **ASPLinux** оболочка sh не используется.

Правда, в каталоге /bin можно обнаружить файл /bin/sh, однако это — не более чем символическая ссылка на /bin/bash, который и запускается, если оболочка sh в явном виде запрашивается каким-либо сценарием.

Список других оболочек, входящих в дистрибутив **ASPLinux** можно получить, просмотрев файл /etc/shells командой

```
cat /etc/shells
```

содержание его следующее:

```
/bin/bash2
/bin/bash
/bin/sh
/bin/ash
/bin/bash
/bin/tcsh
/bin/csh
/bin/zsh
```

Файлы /bin/bash2 и /bin/sh — символические ссылки на /bin/bash. Оболочки ash и zch принадлежат к семейству Shell-совместимых, csh и tcsh — к C-совместимым оболочкам.

Оболочка ash — самая маленькая и компактная из всех используемых в Linux: размер ее около 64 Кбайт (для сравнения, bash — более 500 Кбайт). Этим обусловлены многие ограничения ее возможностей (нет автодополнения, истории команд и ряда других интерактивных возможностей). Однако, в отличие от sh, ash поддерживает некоторые современные функции работы со сценариями. И потому используется главным образом при создании загрузочных дискет для аварийных ситуаций.

В отличие от ash, zsh — одна из самых полнофункциональных оболочек, поддерживающая все современные их функции, как интерактивные (редактирование командной строки, автодополнение, историю команд и т.д.), так и ориентированные на использование в сценариях, а также наибольшее количество команд (более 80) и опций приглашения командной строки (более 50).

Высказывалось мнение, что zsh аккумулирует полезные свойства всех других оболочек. Любознательным пользователям предоставляется возможность самостоятельно проверить справедливость этого утверждения.

Оболочка csh — первый представитель семейства C-совместимых оболочек. По сравнению с близкой по возрасту sh она поддерживала многие дополнительные возможности, ставшие позднее стандартными для современных оболочек (автодополнение, историю команд и т.д.), однако реализованы они обычно иначе, чем в bash. Так, для автодополнения используется не клавиша табуляции Tab, а Esc.

Как и sh, собственно исходная оболочка csh в **ASPLinux** не используется — файл /bin/csh представляет собой символическую ссылку на /bin/tcsh. Оболочка tcsh — клон csh, наделенный многими дополнительными возможностями, реализация которых приближена к таковой bash. При интерактивной работе заметить различие между tcsh и bash очень трудно — оно проявляется только при настройке и создании сценариев, что связано с отличием синтаксиса языка интерпретатора csh. Так, внутренние переменные оболочки tcsh задаются с помощью конструкции вида:

```
set имя [= значение]
```

Для определения переменной окружения используются следующие конструкции

```
setenv  EDITOR  joe
```

и т.д., чем достигается несколько больший (по сравнению с Shell-совместимыми оболочками) лаконизм сценариев.

В заключение следует подчеркнуть, что ни одна из полнофункциональных оболочек, включенных в состав **ASPLinux**, не имеет принципиальных преимуществ перед другими. И использование какой-либо из них — исключительно дело вкуса и привычек пользователя.

Для смены оболочки можно использовать команду `chsh` (от `change shell`). В ответ на нее предлагается сначала ввести пароль пользователя, а затем — указать полный путь к исполнимому файлу новой оболочки

```
New shell [/bin/bash]:
```

Например, для смены `bash` на `zsh`, в этой строке следует ввести `/bin/zsh`, и так далее.

Глава 4

Особенности работы в графическом режиме

ОС Linux, как и все UNIX и UNIX-подобные системы, изначально была ориентирована на работу в текстовом режиме. Однако и графический режим Linux предоставляет большие возможности. Тем более что в Linux пользователь не привязан к одному графическому интерфейсу, как в Windows.

Графический режим Linux реализуется с помощью X Window System, вернее, ее свободно распространяемой реализации — `XFree86`. Именно она включена в состав дистрибутива **ASPLinux**, где представлена в последней своей версии — 4.3.0.

X Window System построена по модели клиент/сервер. В качестве сервера рассматривается аппаратно-зависимая система ввода/вывода, взаимодействующая с видеосистемой (монитором и видеокартой), клавиатурой и мышью. В состав клиентской части входят программы, обеспечивающие прием данных с устройств ввода и вывод их на экран монитора, а также управление интерфейсом (окнами и элементами их управления, меню и т.д.).

Запуск собственно X-сервера осуществляется вводом в командной строке команды `X`, обеспечивающей загрузку графического режима. Однако никакие действия после этого невозможны — для их осуществления требуется хотя бы одна программа-клиент. Важнейшими из них являются эмуляторы терминала, оконные менеджеры и интегрированные среды графического режима. Загрузка их обеспечивается различными стартовыми файлами, о которых (как и о настройке X Window System вообще) рассказывается в руководстве администратора.

Запустить X-сервер с единственной клиентской программой `xterm` (наиболее распространенный из эмуляторов терминала) можно командой

```
xinit
```

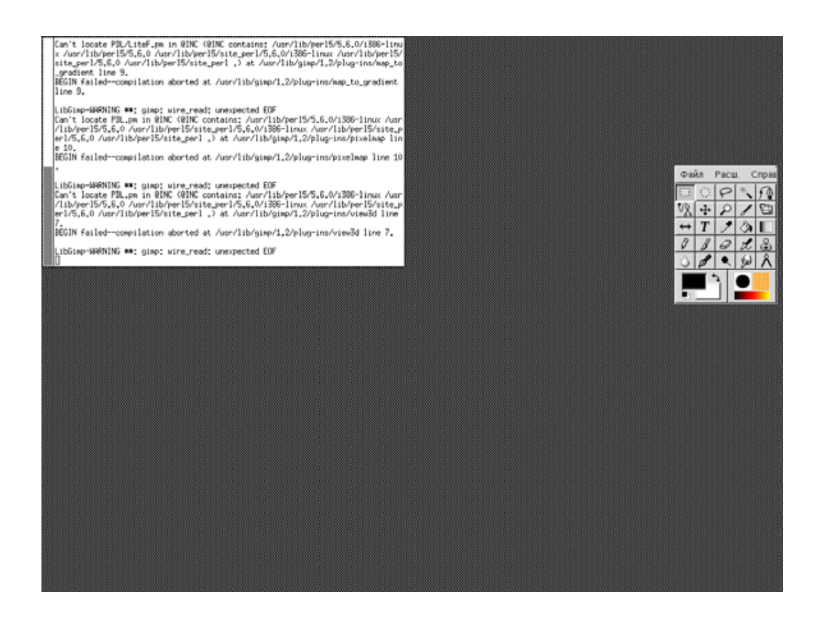


Рис. 4.1: X-сервер с запущенным окном эмуляции терминала

В этом случае происходит загрузка графического режима, в котором появляется окно эмулятора терминала. Оно не имеет никаких управляющих элементов (рис. 4.1), не может быть масштабировано, перемещено или свернуто. Однако в командной строке терминала можно запускать любые программы, точно так же, как это делается в командной оболочке (экземпляр которой, кстати, и запускается в окне терминала). За тем исключением, что в последней могут быть запущены только программы текстового режима, в терминале же — и программы, требующие режима графического.

Как и в консольном режиме, в окне терминала доступна многозадачность — запуск программ (в том числе и других экземпляров `xterm`) в фоновом режиме и контроль заданий. Однако управление запущенными приложениями не очень удобно. Нет даже эффективной возможности переключаться между окнами: окно активизируется, когда курсор помещается в его пределы. И если исходное окно терминала будет полностью перекрыто окнами позднее запущенных программ, получить к нему доступ для завершения или приостановки какой-либо из них не удастся.

Выход из командной оболочки в исходном окне терминала (командой `exit`, но не `logout`) означает одновременно и выход из сеанса X Window System. При этом окна всех запущенных приложений также будут закрыты, и все не сохраненные в них данные — потеряны без предупреждающих сообщений.

Если же доступа к исходному окну терминала получить не удается, сеанс X Window System может быть принудительно завершен комбинацией клавиш `Alt`+`Ctrl`+`Backspace`. Разумеется, несохраненные данные постигнет та же судьба, что и в предыдущем случае.

Может показаться, что использование X Window System при его запуске через `xinit` не очень эффективно. В большинстве случаев это действительно так, хотя иногда такой прием вполне оправдан: например, если требуется запустить всего одно, но ресурсоемкое приложение. В частности, такой минимизированный режим можно применять совместно с интегрированным офисным пакетом StarOffice, который имеет собственные средства управления элементами интерфейса, запуска приложений и т.д.

Кроме того, в командной строке терминала можно запустить любую из клиентских X-программ, предназначенных для создания интерфейса с X-сервером — оконный менеджер или интегрированную среду графического режима. Обычно запуск таких программ включается в стартовый сценарий, выполняемый при запуске команды `startx`.

Команда `startx` запускает X-сервер одновременно с назначенной по умолчанию программой управления интерфейсом — оконным менеджером или интегрированной средой. В **ASPLinux** на стадии установки в качестве такой программы предопределена интегрированная среда *GNOME*. Если пользователь при выборе пакетов отказывается от ее установки, средой по умолчанию становится *KDE*, если же и она не устанавливается — оконный менеджер *TWM*.

Однако пользователь отнюдь не обречен на работу в одной и той же среде.

Во-первых, среду или оконный менеджер очень легко изменить (через менеджер сеансов **GDM**).

Во-вторых, возможен запуск второго (и любого последующего — это лимитируется числом неиспользованных виртуальных консолей и ресурсами компьютера) сеанса X Window System, со своей программой управления интерфейсом.

Правда, воспользоваться для этого командой `startx` уже не удастся: она попытается загрузить ту же среду *GNOME*, а две копии интерфейсной программы одновременно функционировать не могут. И потому следует вернуться к команде `xinit` с дополнительными параметрами.

Так, если первый сеанс X Window System запущен штатной командой `startx`, загрузившей по умолчанию *GNOME* в 7-й виртуальной консоли, следующий X-сеанс можно открыть командой

```
xinit -- :1
```

где `:1` — номер графического дисплея (дисплей первого сеанса неявным

образом получает номер 0). После этого в окне его терминала командой `startkde` можно запустить среду *KDE*. Третий сеанс открывается командой

```
xinit -- :2
```

В нем может быть запущен какой-либо оконный менеджер, например, *WindowMaker* (командой `wmaker`), *IceWM*, *FVWM* и т.д. (каждый из перечисленных — одноименной командой). Если в качестве среды по умолчанию была установлена *KDE*, среда *GNOME* может быть запущена в окне терминала командой `gnome-session`.

Таким образом, пользователь, не меняя настроек по умолчанию, может ознакомиться с возможностями всех имеющихся интегрированных сред и оконных менеджеров и выбрать себе подходящую рабочую среду.

В принципе большинство развитых оконных менеджеров обеспечивает базовый набор возможностей управления интерфейсом — запуск приложений (стартовые меню, панели запуска, строки минитерминала, контекстное меню), переключение между ними (панели задач и т.д.), управление поведением окон (их активизацией, масштабированием, перемещением и закрытием) и рабочим пространством (виртуальные рабочие столы). Однако в одних из них таких возможностей больше, в других — меньше. Кроме того, различия между оконными менеджерами заключаются в возможностях настройки и ее методах (интерактивно или редактированием файлов конфигурации).

Интегрированные среды, помимо управления интерфейсом (осуществляемым средствами встроенного, как в *KDE*, или подключаемого внешнего, как в *GNOME*, оконного менеджера), предоставляют дополнительные возможности — единые средства конфигурирования и более или менее полные и развитые комплекты утилит и приложений.

Однозначные рекомендации по выбору рабочей среды графического режима дать невозможно. Это определяется привычками и потребностями пользователя, ресурсами его компьютера и, наконец, просто вкусом. Мы надеемся, что описания интегрированных сред и нескольких наиболее распространенных оконных менеджеров, приводимые в следующих главах, помогут сделать выбор.

Глава 5

Интегрированная среда *KDE*

Описание рабочих сред графического режима целесообразно начать с *KDE*.

В настоящий момент это наиболее развитая и функциональная из интегрированных сред. Она располагает наиболее широким спектром специально написанных для нее программ общего назначения, включая собственный комплект офисных приложений. Она отличается гибкостью и простотой настроек, осуществляемых встроенными средствами и не требующих редактирования конфигурационных файлов. И, наконец, *KDE* внешне и по приемам работы не покажется непривычной опытному пользователю Windows.

Основными элементами интерфейса *KDE* являются (рис. 5.1) меню рабочего стола вверху экрана (по умолчанию скрыто), панель, объединяющая функции запуска программ и переключателя запущенных приложений, внизу, и собственно рабочий стол.

Меню рабочего стола имеет следующие пункты: *«Файл»*, *«Новый»*, *«Закладки»*, *«Рабочий стол»*, *«Окна»*, *«Помощь»*. Содержание их в большинстве случаев понятно, поэтому прокомментируем его лишь вкратце.

Пункт *«Файл»* позволяет выполнить команду (через вызов строки минитерминала, о которой подробнее будет сказано ниже), запереть (то есть блокировать) экран и выйти из *KDE* (рис. 5.2).

С помощью пункта *«Новый»* (рис. 5.3) можно создать каталог, HTML-файл или текстовый, а также ссылку на устройство (CD-ROM, флоппи-дисковод и т.п.), на приложение (то есть пиктограмму запуска его на рабочем столе) или на URL (помимо стандартных протоколов здесь подразумевается любой протокол, который поддерживает браузер **konqueror**).

Содержание пункта *«Закладки»* (рис. 5.4) аналогично таковому любого современного браузера демонстрирует закладки, созданные в программе **konqueror**, подробнее о которой будет рассказано в одной из следующих глав.

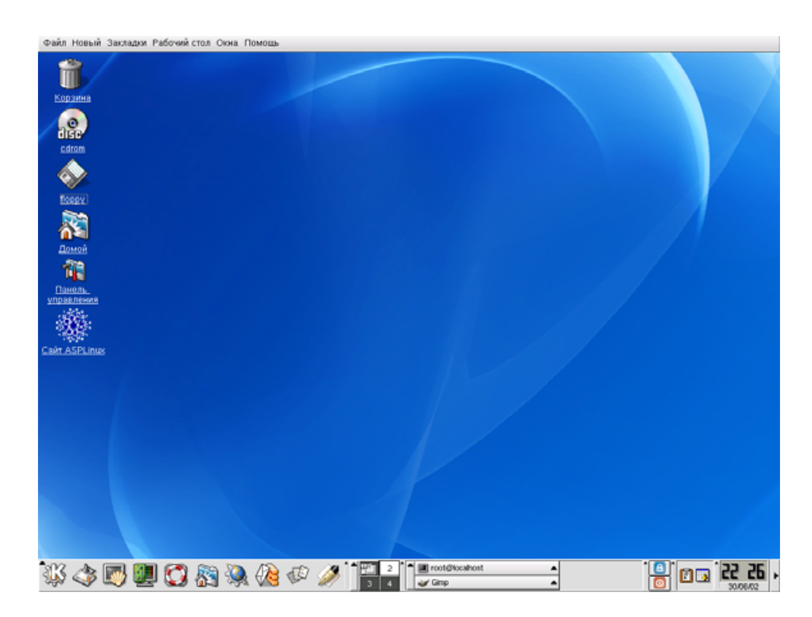


Рис. 5.1: Интегрированная среда *KDE*

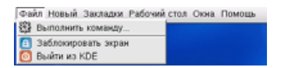


Рис. 5.2: Меню рабочего стола, пункт *«Файл»*

Пункт меню *«Рабочий стол»* (рис. 5.5) позволяет упорядочить окна равномерно или каскадом (эти действия дублируются далее и в меню *«Окна»*), выровнять или упорядочить пиктограммы, отключить меню рабочего стола. Здесь же — настройка фона и настройка собственно рабочего стола. Пункт *«Окна»*, кроме упорядочения окон, предназначен для переключения между открытыми приложениями и виртуальными рабочими столами

В пункте *«Помощь»* вызывается очень подробная и хорошо структурированная справочная система, правда, большей частью на английском языке (рис. 5.6).

Панель *KDE* включает (слева направо): Стартовое меню **К** (функциональное как и **Пуск** в Windows), серию кнопок быстрого запуска приложений (по умолчанию — терминала *KDE*, Центра управления, менеджера файлов, компонентов офисного пакета **KOffice**, справочной системы), переключатель рабочих столов, панель запущенных приложений, системные часы, щелчок мышью на которых вызывает календарь.

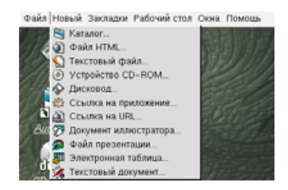


Рис. 5.3: Меню рабочего стола, пункт *«Новый»*

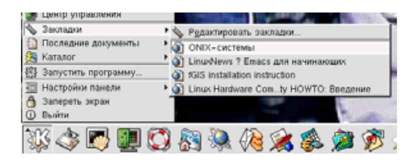


Рис. 5.4: Меню рабочего стола, пункт *«Закладки»*

В состав стартового меню **K** (рис. 5.7) включены в первую очередь собственные программы *KDE*, разделенные на группы, назначения которых ясны из названий:

- Приложения,
- Редакторы,
- Графика,
- Интернет,
- Мультимедиа,
- Офис,
- Настройки,
- Системные,
- Развлечения,
- Утилиты.

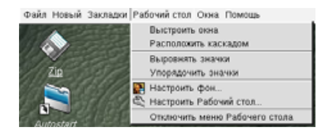


Рис. 5.5: Меню рабочего стола, пункт *«Рабочий стол»*

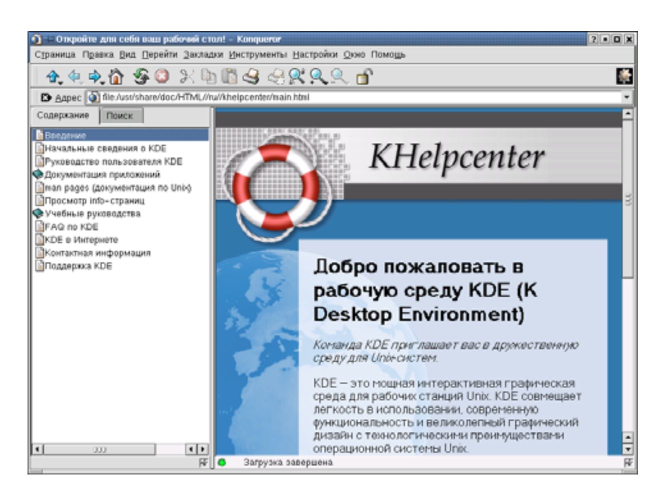


Рис. 5.6: Справочная система *KDE*

Кроме того, приложения, которые являются добавочными для *KDE*, выделены в отдельные подпункты *«Дополнительно»* в каждом из пунктов стартового меню. Стартовое меню содержит также ряд дополнительных пунктов для облегчения навигации (*«Закладки»*, *«Каталоги»*, *«Поиск файлов»* и т.д.). Отдельным блоком в него автоматически включаются часто используемые приложения.

Рабочий стол содержит иконки для быстрого (одним щелчком мыши) запуска приложений и автомонтирования устройств для работы со сменными накопителями (дискетами и дисками CD-ROM). По умолчанию установлено четыре виртуальных рабочих стола, переключение между которыми осуществляется с главной панели.

С рабочего стола доступно контекстные меню. По умолчанию (контекстные меню, как и все остальные интерфейсные элементы *KDE*, легко переопределяются), по щелчку средней кнопкой мыши вызывается аналог пункта *«Окна»* из главного меню (рис. 5.8), по щелчку правой кнопкой — обобщение всех остальных его пунктов (создание файлов и каталогов, настройки и т.д., вплоть до выхода из *KDE*, рис. 5.9).

Контекстные меню (по щелчку правой кнопкой мыши) доступны и для других элементов интерфейса: пиктограмм на рабочем столе, главной панели, отдельных кнопок на ней. В первом случае меню содержит стандартные операции — вырезание, копирование, удаление, помещение в корзину, а также пункт *«Свойства»*. Для иконок сменных накопителей имеются пункты *«Монтировать/Размонтировать»* и (для дисков CD) — *«Извлечь»* (рис. 5.10). Контекстные меню панели и ее кнопок предназначены для конфигурирования и

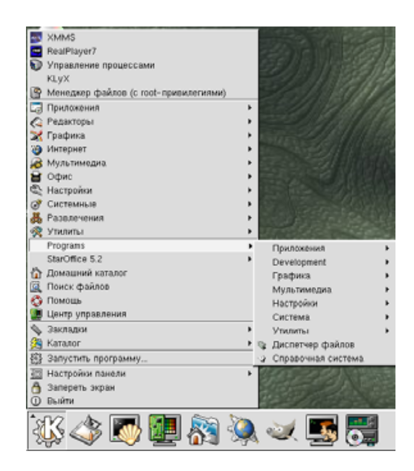


Рис. 5.7: Структура стартового меню *KDE*



Рис. 5.8: Контекстное меню рабочего стола по щелчку средней кнопкой мыши

будут рассмотрены ниже.

Следующий компонент интерфейса *KDE* — окна приложений и элементы их управления. Их можно рассмотреть на примере окна штатного эмулятора терминала **konsole** (рис. 5.11). По умолчанию окно это обрамляется:

- сверху — титульной строкой,
- сразу же ниже ее — строкой меню,
- снизу — инструментальной панелью,
- справа — полосой прокрутки.

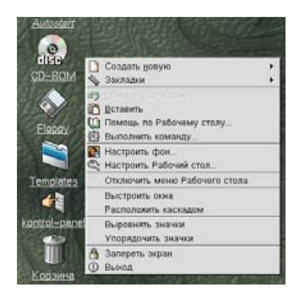


Рис. 5.9: Контекстное меню рабочего стола по щелчку правой кнопкой мыши

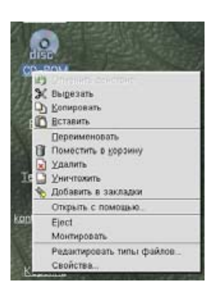


Рис. 5.10: Контекстное меню для пиктограмм рабочего стола (на примере иконки для диска CD)

Все элементы обрамления окна, кроме титульной строки, могут меняться от приложения к приложению. В титульной строке, кроме собственно заголовка, имеются управляющие пиктограммы. Крайнее левое положение в строке (левее заголовка, рис. 5.12) занимает пиктограмма, вызывающая меню управления общими параметрами окна. Пиктограмма, как и прочие пиктограммы титульной строки, реагирует одинаково на нажатие любой кнопки мыши. Пункты меню управления позволяют переместить, свернуть (в пиктограмму или до строки заголовка) и распахнуть окно, произвольно изменить его размер, отправить окно на другой виртуальный рабочий стол или закрыть (вместе с запущенной в нем программой). Можно также изменить стиль оформления индивидуального окна, не затрагивая настроек *KDE* в целом.

Правее расположена пиктограмма, называемая «шпилькой». Она предназначена для фиксации окна в определенном месте рабочего стола.

Справа от заголовка в строке титула — три пиктограммы (слева направо) следующего назначения:

- сворачивания в иконку на Панели,

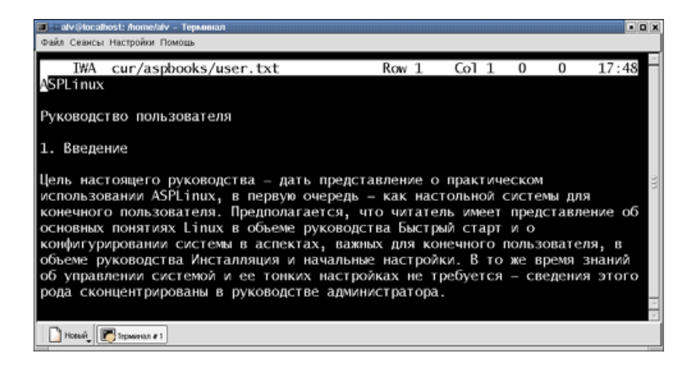


Рис. 5.11: Интерфейсные элементы окна *KDE* (на примере терминала konsole)

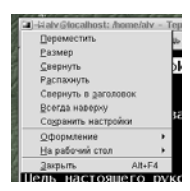


Рис. 5.12: Меню управления окном

- максимизации,
- закрытия (напомним, что последняя пиктограмма, как и прочие, срабатывает при нажатии любой кнопки мыши, и окно может закрыться без всяких предупредительных сообщений).

Средства запуска программ в *KDE* чрезвычайно разнообразны. Во-первых, это традиционный способ запуска из командной строки окна эмуляции терминала. В качестве терминальной программы по умолчанию (вынесенной в панель задач) здесь выступает **konsole**. Но, разумеется, можно использовать и любые другие эмуляторы терминала.

Второй способ запуска программ — так называемый минитерминал (**minickli**, рис. 5.13). Это панель, включающая нечто вроде командной строки, вызываемая многообразно:

- из меню рабочего стола Файл (Выполнить команду),
- из стартового меню (Запуск),
- из контекстного меню Рабочего стола по правой клавише мыши (Выполнить команду).

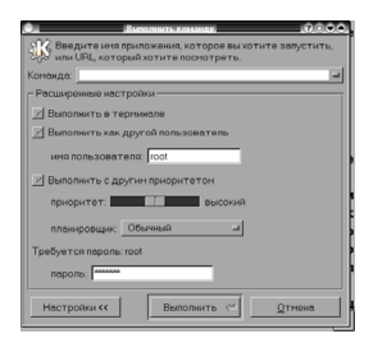


Рис. 5.13: Панель минитерминала minickli

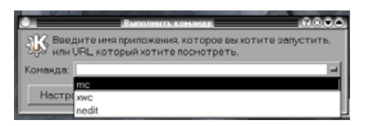


Рис. 5.14: Строка минитерминала; доступ к истории команд — из выпадающего меню

В строке минитерминала можно запустить на исполнение любую программу графического режима, с любыми параметрами. Поддерживается история команд; при этом доступ к ней — не только с помощью стрелок управления курсором `Up` и `Down`, но и из выпадающего меню (рис. 5.14).

Панель минитерминала может использоваться и для запуска программ текстового режима, для этого требуются вполне очевидные настройки. А именно, следует включить переключатель Выполнять в окне терминала. Более того, здесь же можно получить доступ к командам, выполняемым от лица другого пользователя, в том числе и системного администратора (при условии знания соответствующих паролей).

Наряду с командными способами запуска программ в *KDE* имеются и визуальные: через стартовое меню, через панель и, наконец, с помощью пиктограмм рабочего стола.

Кроме средств запуска программ, важно и переключение между ними после запуска. В *KDE* для этого используются:

- традиционная комбинация клавиш ALT+TAB,
- контекстное меню по щелчку средней клавиши мыши,
- пункт Окна из меню рабочего стола,
- правая часть главной Панели, где помещаются иконки-кнопки выполняе-

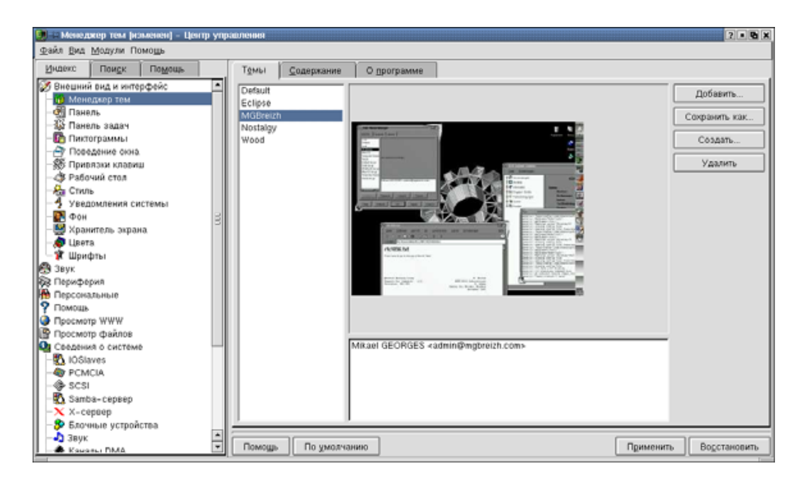


Рис. 5.15: Центр управления *KDE*

мых программ.

Возможности настройки *KDE* практически безграничны: почти любое конфигурирование можно выполнить многими различными способами. Так, настройку элементов интерфейса (рабочего стола, панели и ее отдельных компонентов, пиктограмм и прочего) можно выполнить с помощью контекстного меню, доступного по щелчку правой кнопкой мыши. Однако главным средством конфигурирования *KDE* является его Центр управления (рис. 5.15).

Как можно видеть из рисунка, с помощью Центра управления настройке поддается практически все — от конфигурации аппаратных средств до параметров сети. Большинство средств настройки дублируется иными утилитами или более эффективно осуществляется редактированием конфигурационных файлов. И потому ниже будет рассмотрен только специфический раздел Центра управления — Внешний вид и интерфейс (в англоязычном оригинале — LookNFeel), так как настраиваемость интерфейса - одна из самых сильных и привлекательных сторон *KDE*.

Раздел Внешний вид и интерфейс включает следующие пункты:

1. Менеджер тем, позволяющий выбрать стилистически единое оформление всего интерфейса, т.н. тему, из предопределенного набора, который может быть дополнен, видоизменен и сохранен для последующего использования; возможно и создание собственных тем.

2. Панель — дает возможность настройки таких параметров главной Панели, как ее расположение (с любой стороны экрана), размер, автома-

тическое скрытие при потере фокуса, элементы стартового K-меню, а также фоновые изображения для все панели и отдельных ее кнопок.

3. Панель задач — здесь можно включить показ всех открытых окон; по умолчанию на панели задач выводятся только окна, открытые на текущем виртуальном рабочем столе.

4. Пиктограммы — этот пункт определяет общий стиль пиктограмм рабочего стола и их размер (от 16x16 до 48x48 пикселей).

5. Поведение окна — описывает условия фокусировки окон (щелчком мыши, перемещением курсора в пределы окна и т.д.), способ переключения между окнами и виртуальными рабочими столами, события, приписанные кнопкам мыши при щелчке на разных элементах окна.

6. Привязки клавиш — описывают (и позволяют настроить) комбинации клавиш для выполнения различных действий.

7. Рабочий стол — здесь определяется количество виртуальных рабочих столов (вплоть до 16), гарнитура и размер основного системного шрифта, цвет текста и фона. Тут же настройка так называемых активных (или «волшебных» границ рабочего стола, то есть зон по краю экрана, при попадании в которые курсора мыши происходит автоматическое переключение на другой виртуальный стол.

8. Стиль — определяет общий вид всех графических элементов (в стиле NextStep, MacOS и т.д.).

9. Уведомления системы — определяют ее реакцию на различные события: запись в файл, звуковые сигналы, вывод диалогов и т.д.

10. Фон — позволяет задать цвет и заливки, в том числе различно ориентированные градиентные (линейные, радиальные, концентрические и прочие) рабочих столов, назначить им фоновые изображения (т.н. «обои» — `wallpapers`).

11. Хранитель экрана — позволяет выбрать таковой из ряда предопределенных, установить время ожидания и выход из хранителя по паролю;

12. Цвета — определяют цветовую гамму интерфейсных элементов.

13. Шрифты — здесь определяются гарнитуры, кегли и начертания для подписей ко всем интерфейсным элементам: Общий, который одновременно является шрифтом по умолчанию во всех *KDE*-приложениях (если их внутренними средствами не определено иное); Моноширинный; Значок на Рабочем столе — подписи к пиктограммам; Панель инструментов — подписи к кнопкам на ней и имена загруженных приложений на панели задач; Меню — шрифт меню всех запущенных из *KDE* приложений; Заголовок окна — шрифт надписи титульной строки; именно здесь следует производить изменения, если после загрузки не читаются русскоязычные надписи.

Для среды *KDE* написано большое количество приложений, многие из которых включены в дистрибутив **ASPLinux** и будут охарактеризованы ниже.

Глава 6

Интегрированная среда *GNOME*

Интегрированная среда *GNOME* несколько отличается от *KDE*. Она не имеет собственного оконного менеджера, но позволяет подключать многие из них в качестве средств управления интерфейсом. В **ASPLinux** в качестве оконного менеджера *GNOME* по умолчанию используется *Sawfish*. И все сказанное ниже относится к этому случаю. При использовании иного оконного менеджера (допустимы *WindowMaker* и *IceWM*) внешний вид отдельных элементов может быть иным, хотя приемы управления и настройки сохраняются.

Как и в *KDE*, основными элементами интерфейса в *GNOME* являются рабочий стол с пиктограммами запуска программ и панель, совмещающая функции быстрого запуска и переключения между приложениями (рис. 6.1). На панели же располагается и переключатель виртуальных рабочих столов (по умолчанию их четыре, но максимальное количество не ограничено).

Пиктограммы, вынесенные на рабочий стол, позволяют запускать связанные с ними приложения двойным щелчком мыши. Однако основным способом запуска программ (помимо эмулятора терминала, представленного в виде собственного терминала *GNOME*, изображенного на рис. 6.1), является стартовое меню *GNOME* (аналог меню *KDE* и меню Пуск в Windows 9x).

Стартовое меню *GNOME* включает следующие главные пункты (рис. 6.2):

- Программы,
- Избранное,
- Апплеты,
- Меню *KDE*,
- Выполнить,
- Панель,
- Заблокировать экран,

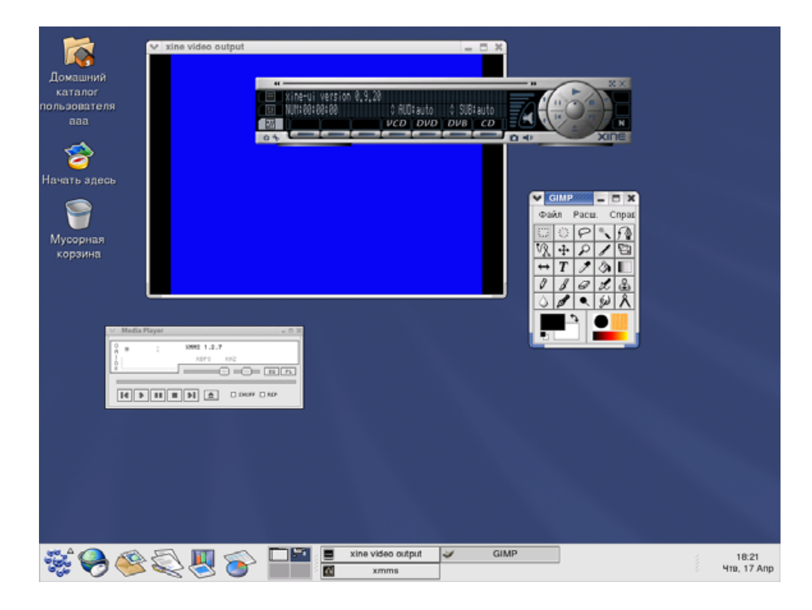


Рис. 6.1: Интегрированная среда *GNOME* — внешний вид

- Выйти из системы.

Наиболее важным (и сложно организованным) является подменю Программы (см.рис. 6.2). Оно включает группы: Приложения, Утилиты, Разработка, Игры, Графика, Интернет, Мультимедиа, Настройки, Система, в которые объединены пункты запуска программ соответствующего назначения. В группе приложений это преимущественно текстовые редакторы и процессоры, в группе Development — средства разработки и т.д.

Пункт Меню *KDE* дублирует стартовое меню этой интегрированной среды, подобно тому, как из *KDE* через пункт K-меню Programms можно получить доступ к меню *GNOME*.

Пункты Выполнить предоставляет доступ к панели минитерминала, позволяющей запускать приложения как графического, так и текстового режимов из командной строки (рис. 6.3). Через выпадающее меню поддерживается также история ранее введенных команд.

Через пункт *«Панель»* осуществляется ее настройка, о чем будет сказано ниже.

Большинство действий по запуску и управлению программами доступно также через контекстные меню рабочего стола. По щелчку на рабочем столе

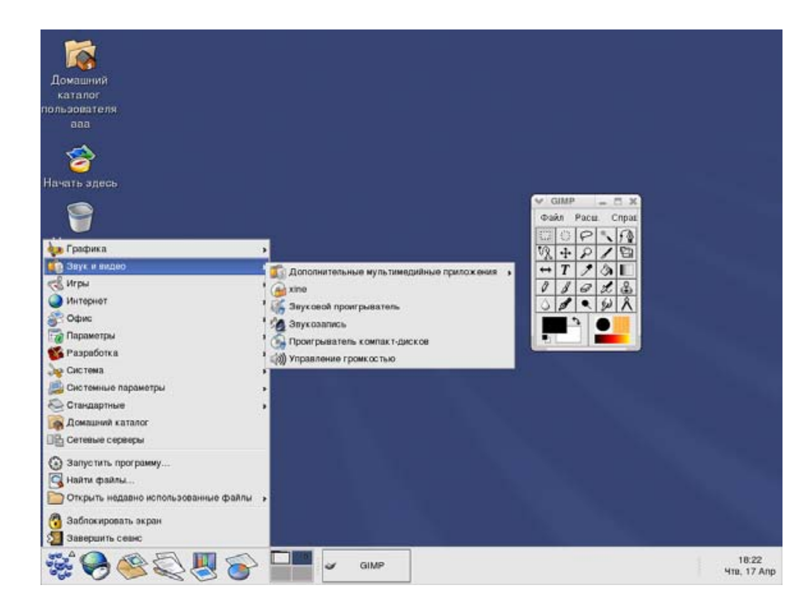


Рис. 6.2: Стартовое меню *GNOME*

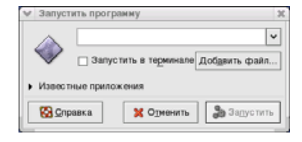


Рис. 6.3: Панель минитерминала *GNOME*

средней кнопкой мыши вызывается меню управления окнами и приложениями, включающее пункты (рис. 6.4):

- Окна, где выводится список всех открытых в данный момент приложений.
- Рабочие столы, позволяющий переключиться на любой из существующих рабочих столов, создать новый стол, а также (уникальная опция *GNOME*) объединить содержание текущего рабочего стола с таковым предшествующего и последующего.
- Программы, дублирующие содержание одноименного пункта стартового меню.
- Настройки, каждый пункт в котором вызывает соответствующий раздел Центра управления *GNOME*.

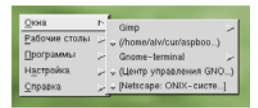


Рис. 6.4: Контекстное меню рабочего стола — управление окнами и приложениями

- Справка, где можно вызвать ответы на часто задаваемые вопросы (ЧаВо) или руководство по *GNOME*, связаться с WWW-сайтом *GNOME* и т.д.

Щелчок правой кнопкой мыши на рабочем столе вызывает меню управления рабочим столом. Здесь можно создать (пункт Новое) окно эмуляции терминала, каталог, ссылку, пиктограмму для запуска, открыть окно файлового менеджера (пункт Создать новое окно), настроить фон и другие свойства рабочего стола.

Элементы управления окном в *GNOME* по умолчанию сходны с таковыми в *KDE*. Кнопка с левого края титульной строки вызывает меню управления текущим окном, позволяющее свернуть, развернуть и закрыть его, переместить на следующий или предыдущий рабочий стол, на передний или на задний план текущего стола и т.д.

Кнопки в правом углу титульной страницы отвечают за (слева направо) минимизацию, максимизацию и закрытие окна.

Таковы особенности интерфейса *GNOME* по умолчанию. Однако все они могут быть настроены в очень широких пределах. Главным средством для этого является Центр управления (рис. 6.5).

Прочие настройки интерфейса *GNOME* (рис. 6.6) во многом производны от свойств выбранного оконного менеджера. Читателю предоставляется возможность изучить их, при желании, самостоятельно.

Как и *KDE*, среда *GNOME* включает большое количество специально созданных для нее приложений, большинство из которых включено в состав дистрибутива **ASPLinux**.

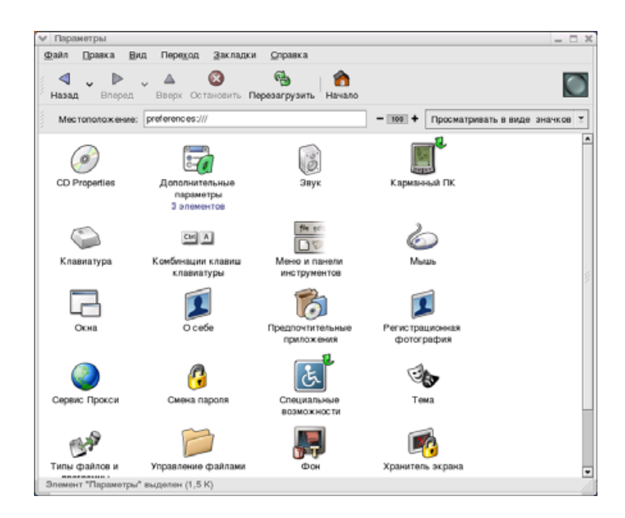


Рис. 6.5: Центр управления *GNOME*

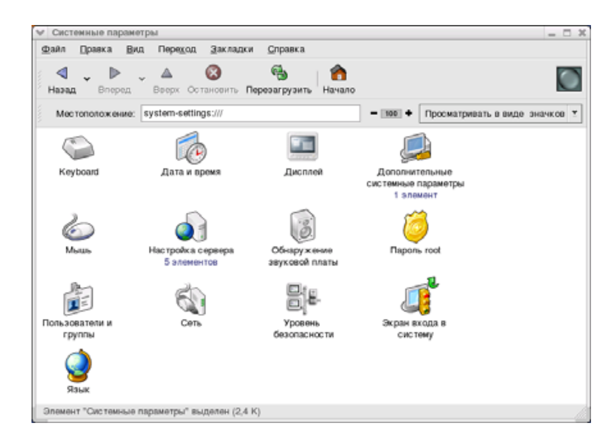


Рис. 6.6: Центр управления: прочие настройки

Глава 7

Оконные менеджеры

В отличие от интегрированных сред, оконные менеджеры предоставляют только средства управления интерфейсными элементами (окнами и их обрамлением, рабочими столами и т.д.), а также более или менее развитые средства запуска приложений и переключения между ними. Компенсацией за это ограничение возможностей являются большее быстродействие, меньшая требовательность к ресурсам, а главное — большая гибкость и широкие возможности настройки, хотя и не всегда просто осуществляемой.

В состав дистрибутива **ASPLinux** включено несколько различных по функциональности и возможностям оконных менеджеров. Остановимся на двух из них, а именно на *IceWM* и *WindowMaker*, представляющих две линии развития оконных интерфейсов — Windows – подобную и продолжающую традиции NextStep.

7.1 IceWM

Оконный менеджер *IceWM* — типичный представитель линии Windows – подобных менеджеров, что подчеркивается его внешним видом (рис. 7.1). В нижней части его экрана можно видеть панель запуска приложений, выполняющую также функции переключателя виртуальных экранов и панели задач. В левой части панели — кнопка вызова стартового меню, правее которой — кнопки для запуска некоторых приложений (терминала, `Mozilla` и т. д.). Далее идут кнопки переключения виртуальных экранов (по умолчанию их четыре). На остальном пространстве панели располагаются кнопки-иконки запущенных приложений.

Стартовое меню (рис. 7.2) по умолчанию содержит пункты для запуска эмуляторов терминала (**xterm** и **rxvt**), **Netscape Navigator** и **GIMP**, группы **Applications**, **System**, **Utilites**. Отдельной группой выделены приложения *GNOME* и переключатель оконных менеджеров: *IceWM* позволяет пе-

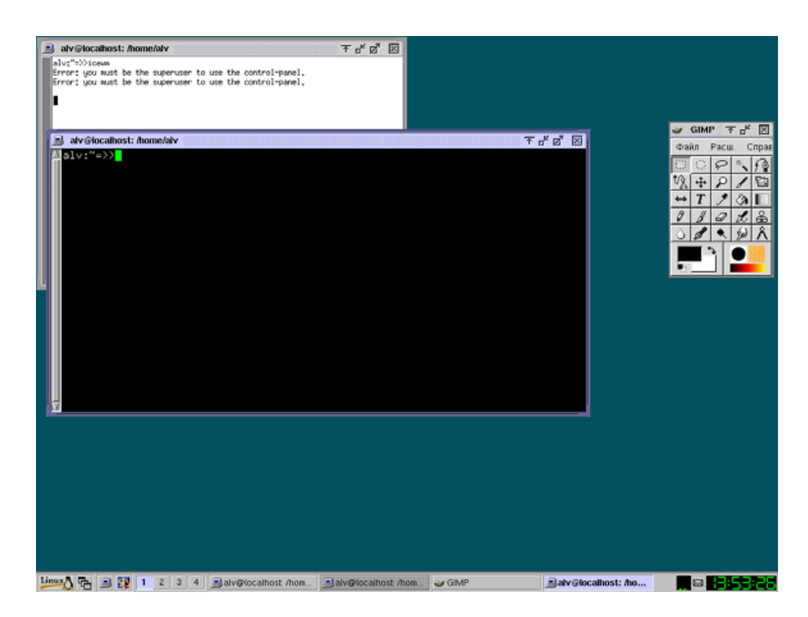


Рис. 7.1: Оконный менеджер IceWM

реключаться (без перезапуска сеанса X Window System) на использование *WindowMaker*, *Blackbox* и *FVWM2*. Кроме того, в стартовом меню есть переключатель тем, о котором будет сказано чуть ниже.

К рабочему столу привязаны три контекстных меню. По щелчку левой клавиши мыши вызывается переключатель виртуальных столов и запущенных приложений (рис. 7.3).

Средняя клавиша открывает **«Window List»** — список открытых окон, также позволяющий переключаться между ними (рис. 7.4).

В отличие от предыдущего, он не исчезает с экрана, занимая положение поверх других окон, пока не будет закрыт явным образом. Наконец, щелчок правой кнопкой мыши вызывает меню приложений, аналогичное стартовому (рис. 7.5).

Таким образом, на первый взгляд в *IceWM* обнаруживается три способа запуска приложений: из стартового и контекстного меню и из окна эмуляции терминала.

Однако есть и четвертый способ, хотя, чтобы прибегнуть к нему, потребуются некоторые описанные ниже настройки (по умолчанию такой терминал включен). Это командная строка минитерминала, включаемая в панель задач

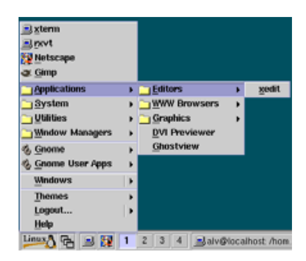


Рис. 7.2: Стартовое меню IceWM



Рис. 7.3: Контекстное меню — переключатель рабочих столов

(рис. 7.6). В нее можно вставить выделенный фрагмент (стандартным способом, средней клавишей мыши) или фрагмент, предварительно скопированный (сочетанием клавиш `Ctrl`+`V`). Разумеется, можно и просто набрать команду с клавиатуры.

Внутренние настройки *IceWM* сводятся к выбору из фиксированного, хотя и обширного, набора тем (рис. 7.7), определяющих цветовую гамму, гарнитуры и начертания шрифтов, а также вид окон и их управляющих элементов. Возможны настройки в стиле OS/2 версий 3 и 4, Windows 95 и многих других.

Для более тонкой настройки *IceWM* необходимо редактирование конфигурационных файлов. Их образцы расположены в каталоге `/usr/share/icewm/`, содержащем четыре файла — `preferences`, `menu`, `toolbar` и `winoptions`. Чтобы сделать их доступными для редактирования пользователем, этот каталог должен быть скопирован в домашний каталог пользователя

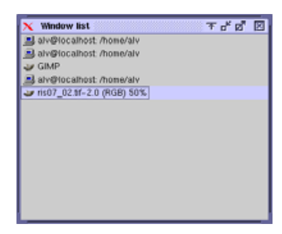


Рис. 7.4: Контекстное меню — Window List

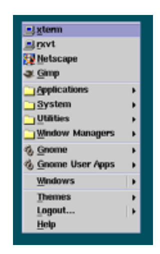


Рис. 7.5: Контекстное меню — меню приложений

и переименован в `.icewm`.

Далее настройке поддается практически все.

Основным конфигурационным файлом является `preferences`. По умолчанию все пункты в нем (а их там чуть ли не две сотни) отключены знаком комментария.

Снимая его и устанавливая переключатели (1/0) в требуемые значения, можно настроить:

- множество параметров активизации окон,
- шрифты и цвета всех интерфейсных элементов,
- установить автоскрытие панели задач и состав иконок на ней (таких, как часы, мониторинг загрузки процессора и так далее), двойную высоту управляющей панели, после чего она разбивается на две части по вертикали; в верхней половине — стартовая кнопка, кнопки запуска приложений и часы, между которыми располагается командная строка мини-терминала (см. рис. 7.6).

В файле menu, как следует из его названия, можно настроить состав стартового меню, вызываемого по кнопке Linux. Делается это достаточно просто: пункт для запуска программы определяется словом prog, после чего следует название приложения, имя иконки и имя исполняемого файла. А чтобы объединить несколько программ в одну папку (то есть сделать иерархическое меню), указывается слово menu, после которого, в фигурных скобках, перечисляются все необходимые имена программ, как это показано ниже:

```
menu Applications folder {
menu Editors folder {
prog fte fte fte
prog vim vim gvim
                prog xemacs xemacs xemacs
```

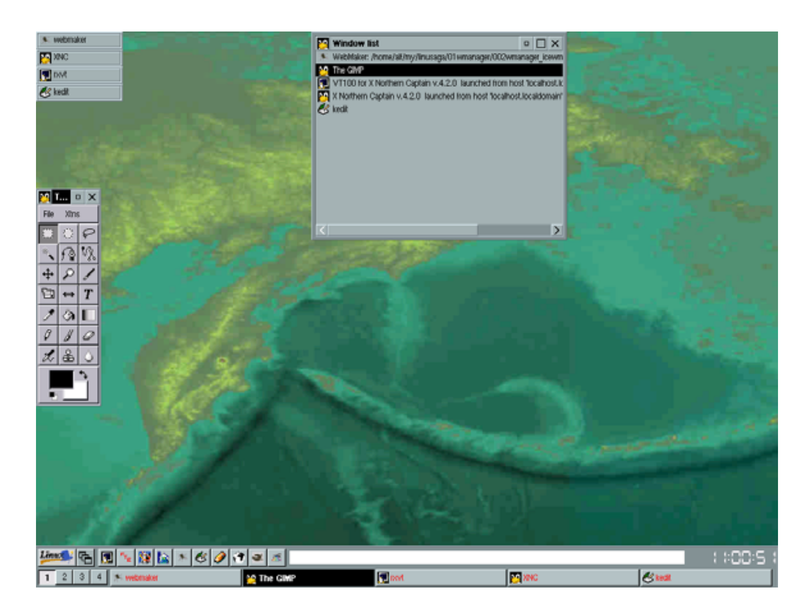


Рис. 7.6: Панель IceWM двойной высоты со строкой минитерминала

```
                prog emacs emacs emacs
                prog xedit xedit xedit
                prog Lyx emacs lyx
                }
}
```

В файле toolbar определяется, какие иконки для запуска приложений будут доступны непосредственно с управляющей панели. Это делается абсолютно так же, как и в настройке стартового меню:

```
prog XTerm xterm xterm
prog FTE fte fte
prog Netscape netscape netscape
```

Наконец, в файле `winoptions` можно настроить разнообразные параметры приложений, в частности, вид пиктограмм для них.

Поддаются ручной настройке и темы рабочего стола. Для этого в каждой теме имеется файл `default.theme` (где задаются цвета, бордюры и прочее) и множество графических файлов для интерфейсных элементов. Не составляет труда изменить цвета любых элементов, шрифты, определить фоновое изображение. Для сохранения темы после окончания текущего сеанса путь к соот-

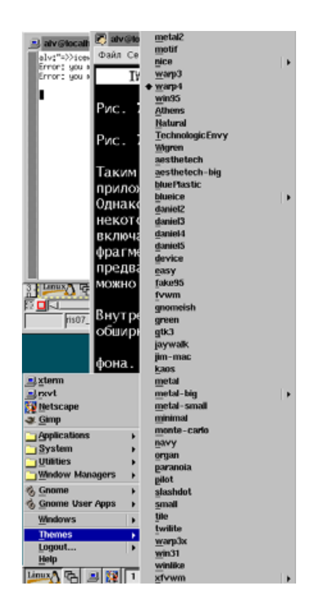


Рис. 7.7: Меню выбора тем рабочего стола

ветствующему файлу `default.theme` нужно прописать в `preferences`. Там же, при необходимости, можно разрешить/запретить центрирование фонового рисунка и его размножение.

Из сказанного можно видеть, что *IceWM* — очень гибкая и настраиваемая система. Разумеется, ручная правка конфигурационных файлов требует некоторого времени. Однако существуют и интерактивные утилиты настройки типа `icewm-pref`, облегчающия процесс настройки (рис. 7.8), но не всегда дающие возможность изменить любые параметры в конфигурационных файлах.

В целом *IceWM* сочетает в себе эффективные средства запуска программ, удобную и разнообразную манипуляцию запущенными приложениями, достаточное количество виртуальных экранов с удобными способами перемещения между ними. Как быстродействие его, так и устойчивость среди всех оконных менеджеров, о которых подробнее несколько ниже, близки к рекордным.

7.2 WindowMaker

Интерфейс *WindowMaker* развивает линию NextStep — знаменитой ОС начала 90-х годов, признаваемой одним из самых удачных дизайнерских решений в

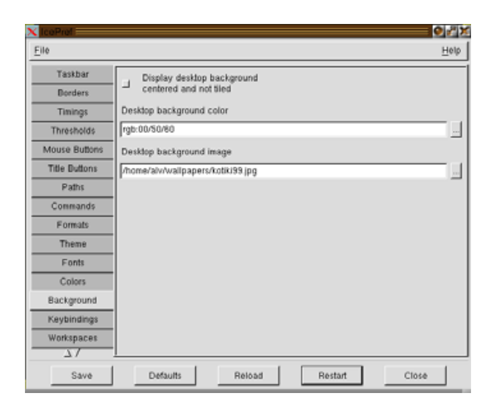


Рис. 7.8: Icewm-pref — утилита интерактивной настройки IceWM

этой области.

Основными элементами интерфейса *WindowMaker* являются управляющая панель WMDock, служащая для запуска приложений, и кнопка-переключатель **«CLIP»** для переключения виртуальных рабочих столов и фиксации окон приложений на рабочем столе (пришвартования, в терминологии программы, рис. 7.9).

Верхняя кнопка управляющей панели (WMDock) по умолчанию ничего не запускает, но служит для управления положением панели на экране: ухватив за нее, панель можно перемещать вверх-вниз или с правой на левую сторону экрана (на верхнюю или нижнюю — нельзя). Однако и к ней можно привязать какое-либо приложение, которое требуется запускать при старте *WindowMaker*.

Вторая сверху кнопка (**«ASCLOCK»**) — это просто индикатор времени и даты. Третья (**«XTERM»**) запускает одноименный эмулятор терминала, четвертая — (**«WMPREFS»**) служит для конфигурирования *WindowMaker*.

Все остальное пространство экрана свободно и представляет собой рабочий стол. Щелчок на нем правой клавишей мыши вызывает ниспадающее меню приложений (рис. 7.10), средней — список открытых окон (рис. 7.11), не исчезающих, в отличие от *KDE* (и Windows) самопроизвольно: чтобы закрыть их, следует щелкнуть соответствующей (правой или средней) клавишей вне меню на рабочем столе.

Из контекстного меню создаются (в любом количестве) и удаляются новые виртуальные экраны (*«Application»- «Workspace»- «Создать»*), между которыми потом можно переключаться указателями на кнопке **«CLIP»**. Переключение экранов сопровождается объемной анимацией их названий в центре

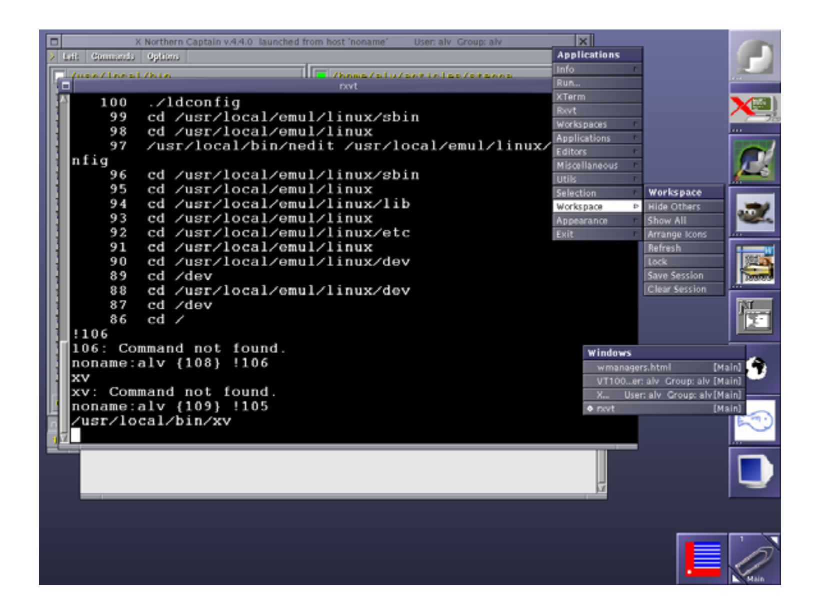


Рис. 7.9: WindowMaker — внешний вид.

экрана. Название экрана легко изменяется щелчком правой клавиши на кнопке **«CLIP»**. Открытые приложения по умолчанию не переходят на новый экран.

Для запуска приложений, кроме кнопок на панели и стандартного терминала, используется также контекстное меню рабочего стола. Для чего в нем присутствует пункт *«Run»*, вызывающий командную строку минитерминала.

Управляющая панель может пополняться кнопками запуска приложений различными способами. Первый — запустить приложение, выбрав из контекстного меню рабочего стола упомянутый пункт *«Run»*. Одновременно с открытием окна приложения на рабочем столе появляется его пиктограмма, внешне аналогичная кнопкам панели. Она захватывается мышью и просто перетягивается в панель.

Для удаления кнопки из панели она просто захватывается мышью и перетягивается за ее пределы на рабочий стол.

Ко второму способу требуется прибегнуть для встраивания приложений *KDE*. Он требует: запустить приложение *KDE*, затем по щелчку правой клавиши мыши на заголовке ее окна вызвать контекстное меню, выбрать в нем пункт Свойства окна, перейти к подпункту Дополнительные параметры и отметить там опцию Эмулировать значок приложения, сохранить эту установку и перезапустить приложение. При следующем его запуске появляется та самая

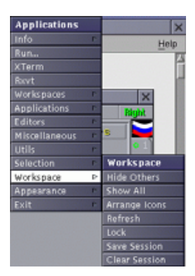


Рис. 7.10: Контекстное меню рабочего стола WindowMaker: запуск приложений



Рис. 7.11: Контекстное меню рабочего стола WindowMaker: переключатель открытых окон

пиктограмма, которую можно перетащить в Dock.

Третий способ — правка вручную конфигурационных файлов, о чем речь пойдет чуть позже.

Окно запущенного приложения по умолчанию имеет в строке заголовка два управляющих значка — минимизации слева и закрытия справа. По щелчку правой клавишей мыши на строке заголовка окно можно развернуть на полный экран, свернуть (то есть минимизировать), втянуть (то есть сократить до строки заголовка), выделить, перемасштабировать/переместить, закрыть (нормальное завершение программы) и убить (аварийное завершение программы, то есть команда *«kill»*).

Доступ к свойствам и параметрам также осуществляется из контекстного меню титульной строки. Свойства — это атрибуты окна, такие как наличие/отсутствие строки заголовка, кнопок закрытия и минимизации, изменяемость размера и прочее, а также исходное рабочее место (программу можно жестко привязать к любому из существующих виртуальных рабочих столов).

Параметры — это положение на рабочем столе (всегда вверху или внизу), а также присутствие везде (как я уже говорил, по умолчанию каждое приложение существует только на том виртуальном экране, на котором оно открыто).

WindowMaker имеет богатые возможности настройки, осуществляемой тремя взаимодополняемыми способами.

Первое средство для этого — уже упоминавшаяся кнопка **«WMPREFS»** на панели управления, которая вызывает окно конфигурирования WindowMaker

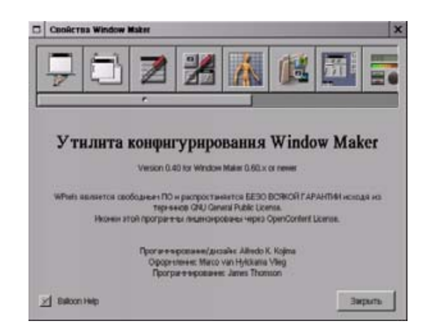


Рис. 7.12: Утилита конфигурирования WindowMaker

(рис. 7.12), где можно настроить:

- позицию открываемых окон (автоматически, случайно, каскадно или вручную);
- фокусировку окна (щелчком или вслед за курсором мыши), а также определить время автоматического всплывания окна (от 10 до 800 мсек, по умолчанию — никогда);
- выравнивание подменю — в стиле Windows, ниспадающим каскадом, или по верхнему краю;
- расположение минимизированных окон и их выравнивание, а также размер экранных кнопок и иконок (от 24x24 до 96x96 пикселей);
- появление и характер всплывающих подсказок (по умолчанию отключены вообще);
- пути поиска графических файлов для пиктограмм и фоновых изображений);
- навигацию по виртуальным экранам (циклическую, с возвратом на первый экран после последнего, или с открытием нового экрана после последнего существующего); здесь же можно отключить WMClip и WMDock (что не рекомендуется — включить обратно их можно только ручной правкой одного из конфигурационных файлов).

Имеются также средства настройки внешнего вида иконок, окон, горячих клавиш, свойств мыши и прочее. А главное — настройка меню приложений, о чем следует сказать подробнее.

В меню *«Приложения»* (то есть *«Applications»*, вызываемое из контекстного меню рабочего стола правой клавишей мыши) можно добавить (и, разумеется, удалить) пункты первого уровня, которые могут содержать подменю

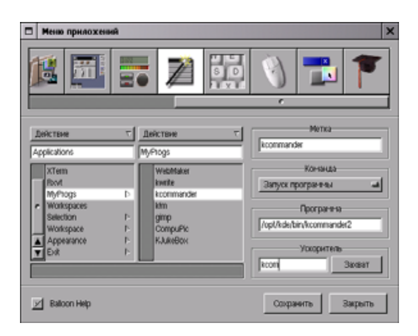


Рис. 7.13: Настройка меню приложений WindowMaker

любого уровня вложенности (рис. 7.13). И с любым количеством пунктов, которым приписываются любые команды — достаточно указать ее имя (или полный путь).

Кроме того, здесь же, кроме выхода из *WindowMaker* и его перезапуска, можно (в подпункте *«Switch to»*) прописать вызов любого доступного оконного менеджера с привязкой к горячей клавише. Делается это абсолютно так же, как и вызов прикладных программ. При этом переключение в другой оконный менеджер или графическую среду (например, *KDE*) происходит с сохранением всех открытых приложений. К сожалению, обратная процедура (то есть, скажем, возврат из *KDE* в *WindowMaker*) невозможна.

Дополнительные настройки внешнего вида *WindowMaker* можно выполнить из всплывающего меню *«Applications»* (подменю *«Appearance»*, а не *«Workspaces»*, как можно было бы ожидать). Здесь можно определить:

- тему целиком (из фиксированного набора, который легко пополнить);
- стили окон и меню (из списка в более чем две дюжины позиций);
- установки иконок;
- фоновое оформление — сплошная или градиентная цветовая заливка, изображения в любом распространенном растровом формате; фон становится общим для всех виртуальных экранов, как существующих, так и создаваемых позднее.

Из комбинации всех этих элементов легко создать собственную тему и сохранить ее в виде файла для дальнейшего использования. Отсутствует только возможность изменения базового шрифта — это делается исключительно вручную.

Кроме этого, *WindowMaker* допускает точную настройку путем редактирования конфигурационных файлов, находящихся в каталоге `/etc/X11/WindowMaker` и (в виде копии) в каталоге `$HOME/GNUstep/Defaults`. Файлов этих пять: `WMGLOBAL`, `WindowMaker`, `WMRootMenu`, `WMWindowAttributes`, `WMState`.

В файле `WMGLOBAL`, как явствует из его названия, определяются наиболее общие параметры. В частности, только здесь можно переопределить шрифт для элементов рабочего стола.

В файле `WindowMaker` указываются стили меню и окон, цветовая палитра, пути для иконок и фоновых рисунков и многие другие параметры, определяемые через WMPrefs, через который их и лучше изменять при необходимости. Однако, если вы случайно отключите WMClip и WMDock, единственный способ вывести их на рабочий стол снова — отыскать в этом файле строки

```
DisableClip = YES;
DisableDock = YES;
```

и заменить `YES` на `NO`. Или просто удалить — значение по умолчанию `NO`. Эта процедура **обязательно** выполняется либо в консольном режиме, либо — в другом оконном менеджере — иначе *WindowMaker* при выходе восстановит установки текущего сеанса.

Файл `WMRootMenu` описывает содержание меню «*Applications*», вызываемого с рабочего стола и также определяемого через WMPrefs. Содержание файла `WMWindowAttributes` ясно из названия. В файле `WMState` дается описание панели WMDock и рабочих столов. Именно его легко отредактировать для внесения новых приложений в WMDock.

В целом *WindowMaker* — весьма быстрый и не очень требовательный к ресурсам оконный менеджер с удобными интерактивными настройками и привлекательным интерфейсом.

7.3 Прочие оконные менеджеры

Кроме этого, в состав **ASPLinux** включено еще несколько оконных менеджеров (обратите внимание, что набор дополнительных оконных менеджерова может меняться от версии к версии):

- *TWM* — простой оконный менеджер с ограниченными возможностями (в частности, без поддержки виртуальных рабочих столов) (рис. 7.14);
- *FVWM2* — один из традиционных для Linux менеджеров окон, отличающийся практически неограниченными возможностями настройки (рис. 7.15);

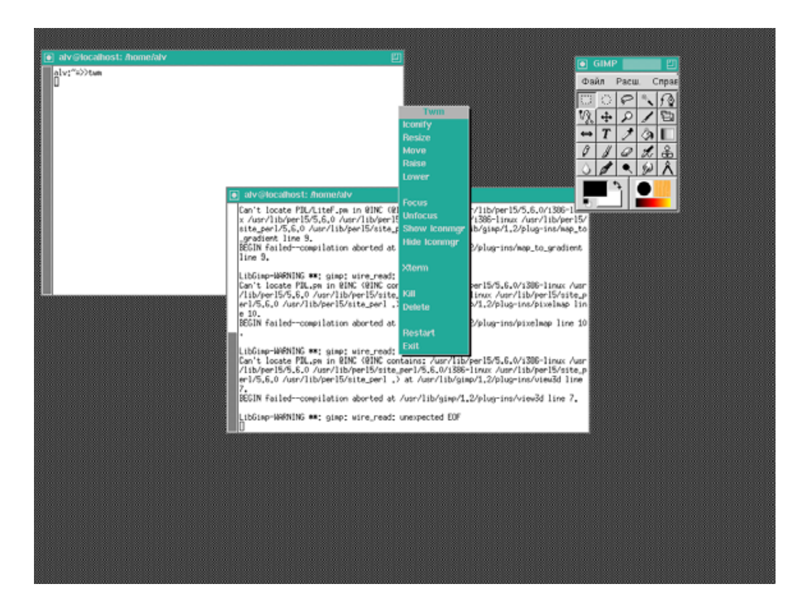


Рис. 7.14: Оконный менеджер TWM

Как и описанные выше, все они обладают своими достоинствами и недостатками.

И потому однозначно рекомендовать какую-либо систему управления интерфейсом затруднительно: пользователь должен оценить их и сделать выбор сам.

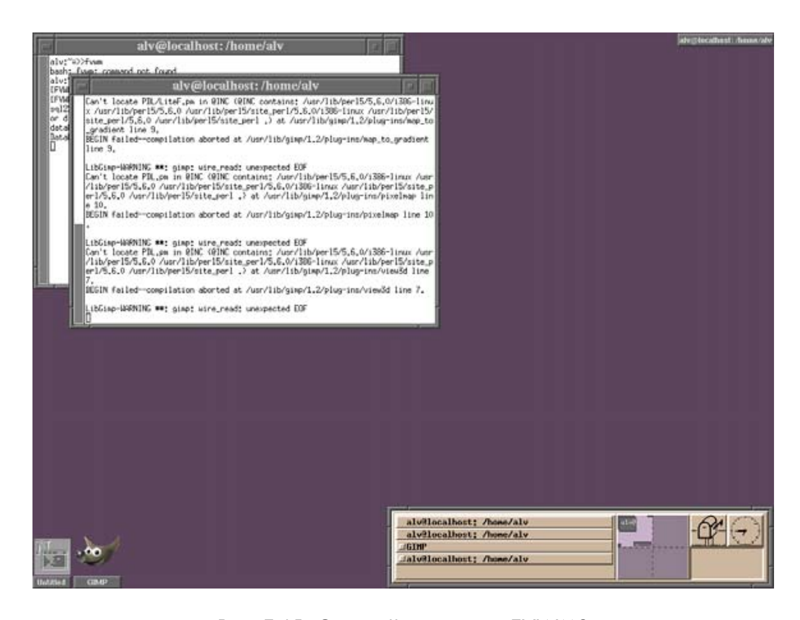


Рис. 7.15: Оконный менеджер FVWM2

Глава 8

Средства работы с файлами и архивами

Управление файлами — их копирование, перемещение, переименование, удаление, поиск и т.д., — задача, возникающая перед пользователем ежедневно. Как было описано выше, один из самых эффективных методов решения этой задачи — использование возможностей командных оболочек (в первую очередь оболочки bash). Однако в **ASPLinux** включены и другие инструменты выполнения файловых операций для текстового и графического режима.

Не менее важны в повседневной работе средства для создания и управления архивами. И здесь также основными являются инструменты командной оболочки.

Наконец, ни в коем случае не следует пренебрегать средствами резервного копирования. Linux вообще (и **ASPLinux** в частности) — система очень надежная и устойчивая. Но от повреждения данных, вплоть до разрушения файловой системы из-за ошибки пользователя или сбоя оборудования, не застрахован ни один пользователь, вне зависимости от квалификации и опыта работы. Кроме того, копирование данных на мобильные носители иногда необходимо для переноса информации между компьютерами.

В этой главе будут рассмотрены средства управления файлами (на примере одной из наиболее эффективных программ — Midnight Commander), средства для архивирования и компрессии данных, и инструменты для создания дисков CD-R/RW. Следующая глава будет целиком посвящена универсальному инструменту для управления файлами как на локальном компьютере, так и в сети — файловому менеджеру и браузеру **Konqueror**.

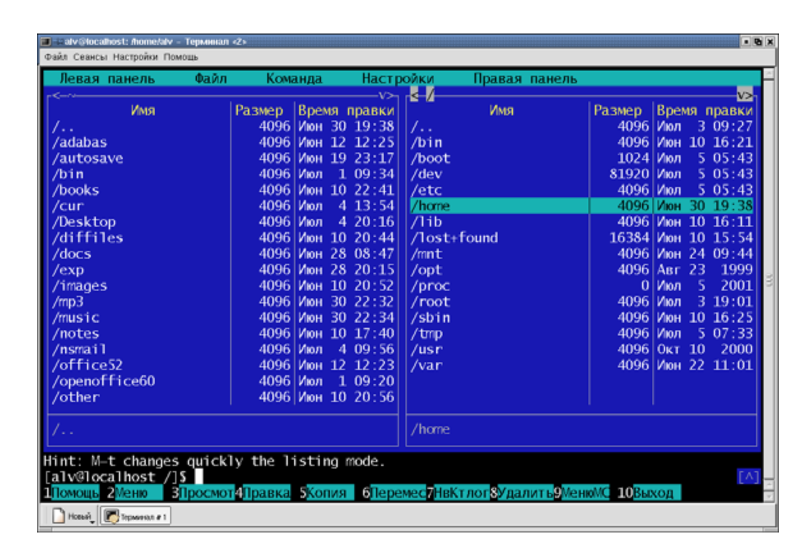


Рис. 8.1: Файловый менеджер Midnight Commander

8.1 Файловый менеджер Midnight Commander

Программа Midnight Commander — это файловый менеджер консольного режима, вызываемый командой mc из командной строки (рис. 8.1).

Основные элементы интерфейса MC — две панели со списками файлов, строка меню в верхней части экрана, панель горячих клавиш в нижней; над последней — командная строка, в которой можно выполнять большинство операций, доступных из оболочки bash.

Файловые операции в MC выполняются двояко — комбинациями горячих клавиш или через меню. Начнем с первого метода, обеспечивающего большее быстродействие — именно ему обязаны своей популярностью клоны Norton Commander.

Прежде всего следует предупредить, что практически рефлекторное действие старого пользователя Norton Commander и его аналогов для MS DOS/Windows — нажатие комбинации Alt+F1/F2 для перехода на другой диск, — в MC к требуемому результату не приведет по понятным причинам. Во-первых, в Linux отсутствует понятие диска в смысле MS DOS: все накопители представляют собой подкаталоги в иерархии файловой системы. Во-вторых, комбинация Alt+F# зарезервирована за операцией перехода на другую виртуальную консоль. И потому не следует удивляться, когда после нажатия Alt+F2 вместо панели выбора диска перед глазами возникает

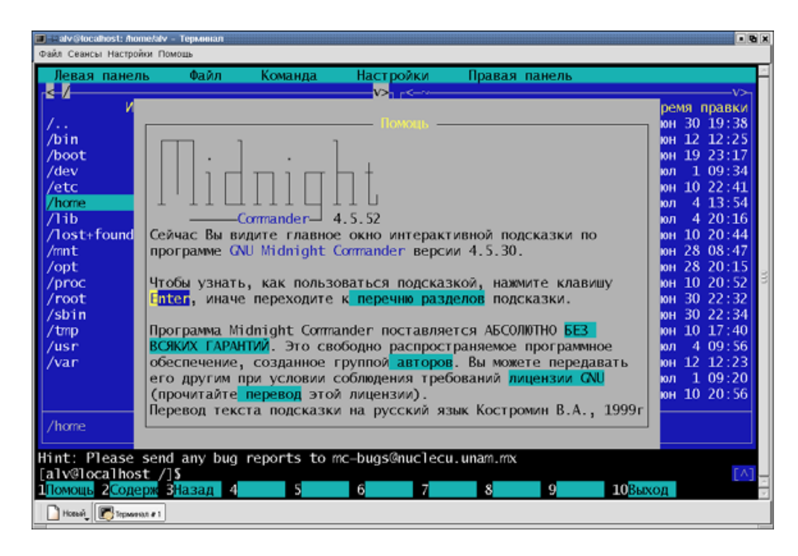


Рис. 8.2: Встроенная система помощи MC

черный экран с предложением авторизоваться. Впрочем, это - чуть ли не единственная сложность в использовании горячих клавиш MC.

Горячие клавиши для наиболее употребимых файловых операций, как уже сказано, вынесены в панель в нижней части экрана. Здесь доступны следующие действия:

- F1 — вызов встроенной системы помощи (рис. 8.2). Кроме этого информацию о MC можно получить с помощью `man mc` или `info mc`;
- F2 — вызов пользовательского меню (рис. 8.3);
- F3 — просмотр содержания текстового файла с помощью встроенной программы. Если текущим является каталог, нажатие этой клавиши приводит к его открытию. Возможен просмотр архивных файлов `*.tar`, в том числе и компрессированных (`*.tar.gz`) — в этом случае нажатие клавиши F3 выводит список файлов архива;
- F4 — редактирование файла во встроенном или внешнем текстовом редакторе;
- F5 — копирование файла, группы файлов или каталога из каталога, выведенного на текущей панели в каталог, открытый на целевой панели. Копирование может быть рекурсивным, то есть охватывать вложенные подкаталоги и входящие в них файлы;

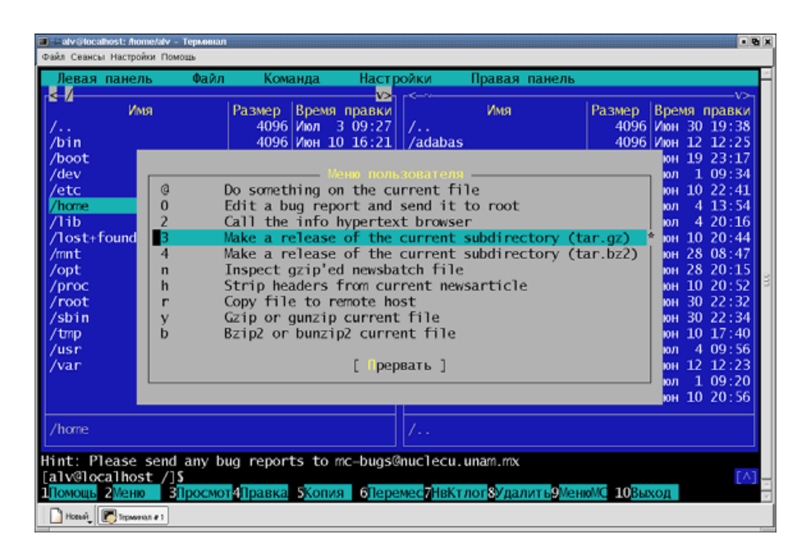


Рис. 8.3: Пользовательское меню MC

- F6 — перемещение и (или) переименование файлов. Для перемещения действуют те же правила, что и для копирования; переименованию поддаются только одиночные файлы или каталоги (рис. 8.4);
- F7 — создание нового каталога;
- F8 — удаление файла, их группы или каталогов, в том числе и рекурсивное;
- F9 — активизация или вызов главного меню;
- F10 — выход из MC, а также прерывание любой ранее начатой операции.

Кроме того, большинство действий выполняемых через главное меню, также дублируется комбинациями горячих клавиш. Однако все они адекватно действуют только в том случае, если MC запущен в консольном режиме. В режиме эмуляции терминала X Window System их поведение зависит от настроек терминала, и часть функциональных клавиш могут не вызывать никаких действий, кроме появления Esc-последовательности в командной строке. Впрочем, часто положение это можно исправить, как будет показано ниже.

Главное меню MC в **ASPLinux** по умолчанию отображается на экране, хотя может быть скрыто соответствующими настройками. Кроме клавиши F9, оно может быть активизировано мышью: в отличие от большинства консольных программ Linux, в MC мышь выполняет роль указательного устройства,

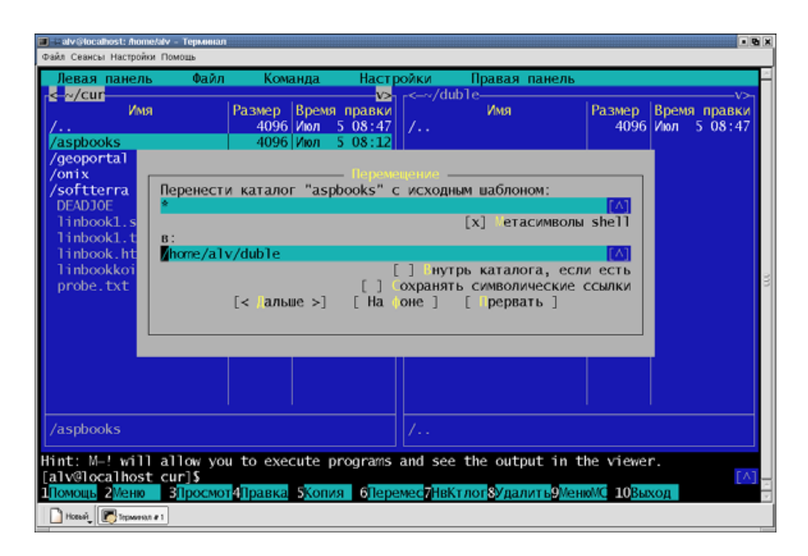


Рис. 8.4: Перемещение/переименование файлов

позволяя выбирать пункты меню, клавиши на панели, файлы и каталоги, которые отмечаются правой кнопкой. Это определяется программой gpm, которая должна быть установлена и включена в число стартовых сервисов. Впрочем, в **ASPLinux** и то, и другое по умолчанию выполнено.

Пункты главного меню следующие:

- Левая (Правая) панель,
- Файл,
- Команда,
- Настройки.

Меню для левой и правой панелей идентичны и позволяют выполнить следующие действия (рис. 8.5):

- определить формат представления списка файлов на текущей панели - стандартный, укороченный, расширенный или определяемый пользователем (рис. 8.6);
- вывести на текущей панели содержание файла, выделенного на противоположной панели (рис. 8.7);

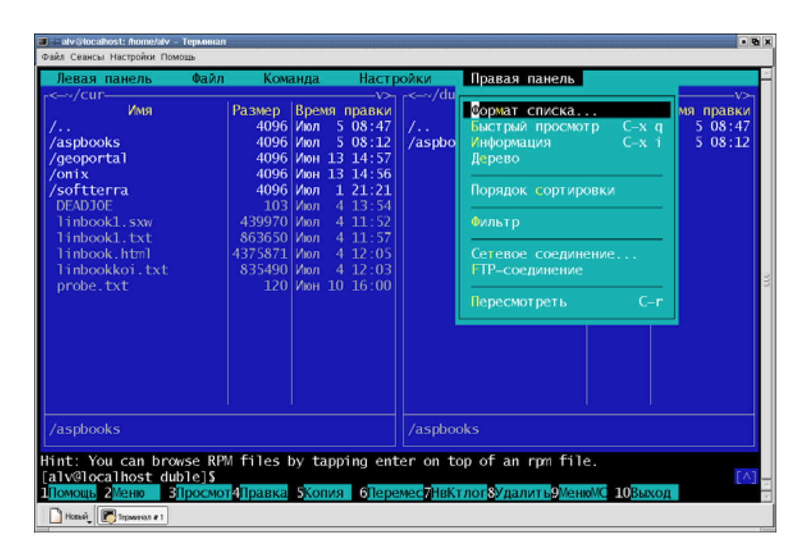


Рис. 8.5: Содержание меню Правой панели

- вывести на текущую панель информацию о файле, выделенном на панели противоположной (рис. 8.8);
- представить содержание текущей панели в виде дерева каталогов (рис. 8.9);
- определить порядок сортировки списка файлов текущей панели (рис. 8.10) — по имени, расширению, времени модификации, времени доступа, времени изменения и т.д.

В этом же меню можно установить соединение с удаленным компьютером по локальной сети и по протоколу ftp. Для этого достаточно просто ввести в соответствующей панели сетевое имя компьютера или ftp-адрес нужного сервера.

Меню Файл включает следующие основные файловые операции (рис. 8.11):

- вызов настраиваемого меню пользователя, предназначенного для наиболее часто используемых действий;
- просмотр файла, отмеченного курсором (аналог клавиши F3) и файла, вызываемого по его имени;
- просмотр команды, например, пользовательского сценария, во встроенном или внешнем редакторе, и запуск ее на исполнение;

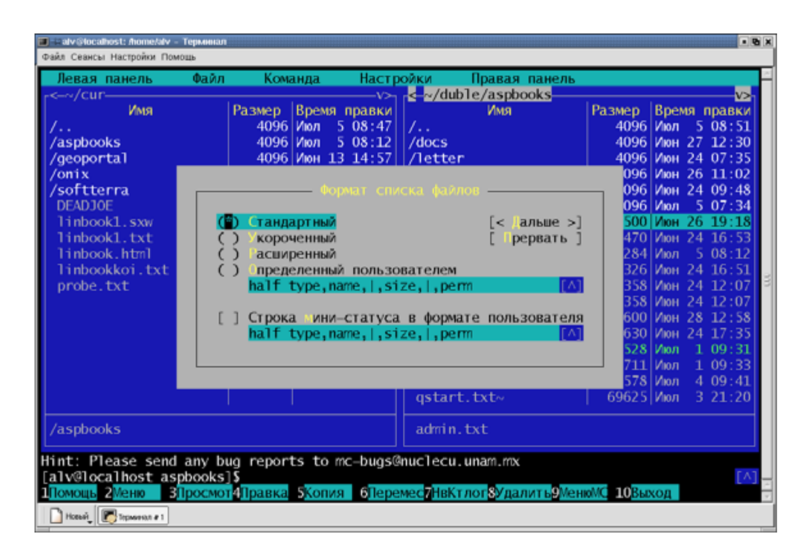


Рис. 8.6: Определение формата списка файлов

- редактирование, копирование, переименование, удаление файлов (аналоги клавиш F4, F5, F6, F8 соответственно), создание каталога (аналог клавиши F7);
- выделение, инвертирование и снятие выделения для группы файлов или каталогов (комбинации клавиш +, * и −, соответственно, на малой цифровой клавиатуре).

Все эти действия подобны таковым в `Norton Commander`. Однако группа пунктов меню Файл отражает специфические для UNIX-систем действия над файлами.

Так, пункт Права доступа определяет полномочия на чтение, запись, исполнение и т.д. (UID, GID) по принадлежности файла — для владельца, группы и прочих (рис. 8.12); права могут быть установлены или изменены сразу для группы выделенных файлов.

Пункты Жесткая ссылка и Символическая ссылка создают файлы-ссылки на выделенный файл, а пункт Правка ссылки позволяет отредактировать уже созданную ссылку.

В пункте Владелец/группа устанавливаются соответствующие атрибуты принадлежности для файла, каталога, группы выделенных файлов (рис. 8.13). В пункте Права (расширенные) права доступа могут быть определены в символьной форме (рис. 8.14).

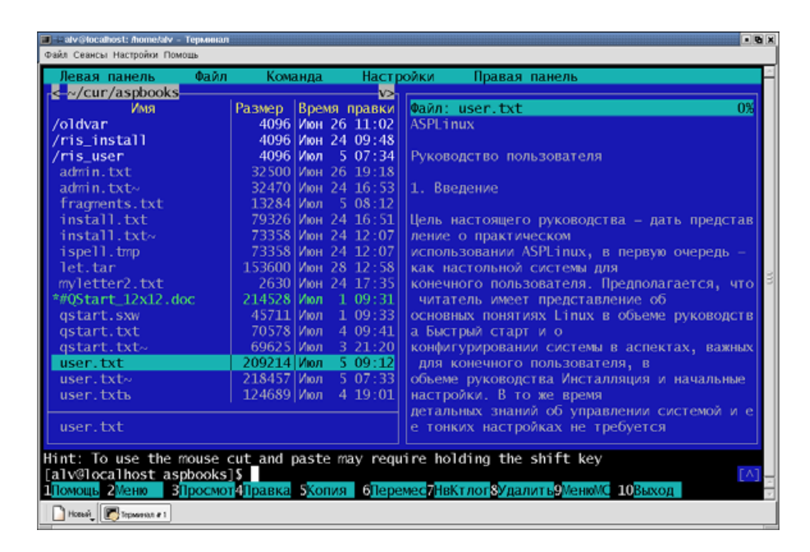


Рис. 8.7: Быстрый просмотр файла

В пункте главного меню Команда (рис. 8.15) осуществляются такие действия, как:

- вывод дерева каталогов, начиная с корневого, для файловой системы (рис. 8.16) с возможностью перехода в отмеченный каталог;
- поиск файла по имени, шаблону, текстовому фрагменту, в том числе и с учетом регистра (рис. 8.17);
- инверсия и отключение панелей, сравнение каталогов, выведенных на панелях;
- вывод в поле Размер полного объема файлов, входящих в состав каталогов (рис. 8.18).

Здесь же можно вызвать историю команд, введенных в течение сеанса MC, просмотреть список фоновых процессов, отредактировать (во встроенном или внешнем текстовом редакторе) индивидуальное меню пользователя (то самое, что вызывается по клавише `F2`).

В меню Настройки можно для начала определить общую конфигурацию MC (рис. 8.19): опции показа, использование встроенного редактора (при отключении используется системный редактор, определенный в переменной окружения `$EDITOR` данного пользователя), автосохранение настроек и т.д.

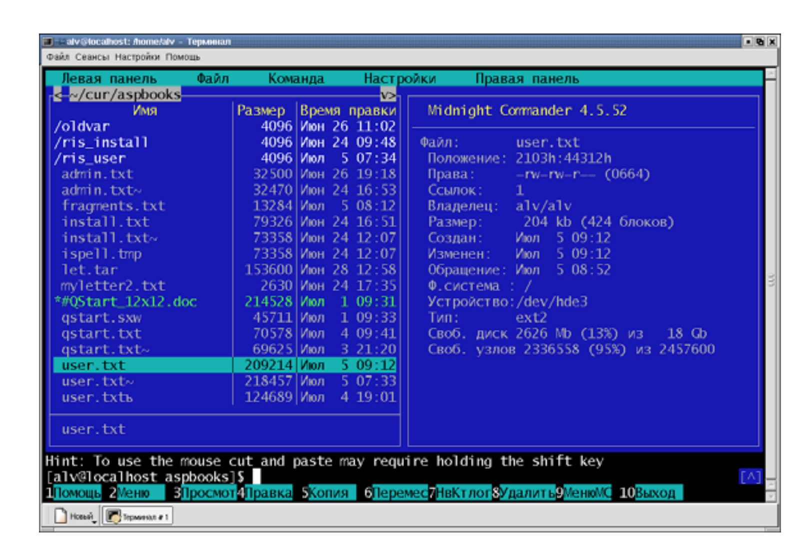


Рис. 8.8: Вывод информации о панели

В пункте Внешний вид (рис. 8.20) устанавливаются:

- разбиение экрана на панели — горизонтально или вертикально, и их относительные размеры;
- выделение цветом — по правам доступа или типам файлов;
- отображение таких элементов интерфейса, как главное меню, командная и статусная строки, панель клавиш, строка подсказки.

Далее, подлежат настройкам подтверждения необратимых действий (рис. 8.21) — удаления и перезаписи, а также исполнения команды и выхода из MC.

Настройка битов символов требуется для корректного отображения имен файлов, передаваемых наборами символов, отличными от чистого (7-битного) ASCII («*Input/display codepage*»), и их восприятия с клавиатуры (полный 8-битный ввод, рис. 8.22).

В частности, здесь можно установить наборы символов для имен файлов, набранных кириллицей — Windows 1251, CP 866, KOI8-R и т.д. (рис. 8.23).

Пункт *«Распознавание клавиш»* (рис. 8.24) может помочь, если при настройке эмулятора терминала в графическом режиме по умолчанию не воспринимаются управляющие клавиши MC. Делается это следующим образом: сна-

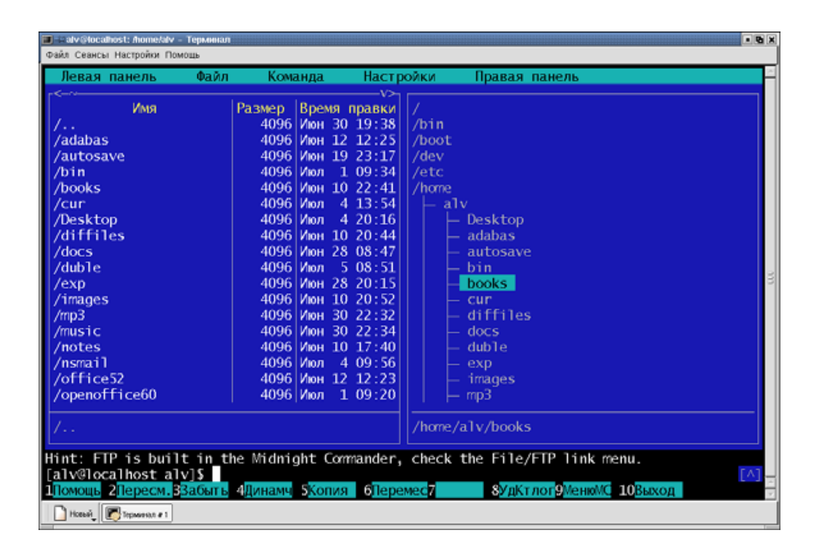


Рис. 8.9: Представление панели как дерева каталогов

чала нужно нажать на все клавиши, выведенные на панели (см. рис. 8.24). Клавиши, нажатия которых обрабатываются MC корректно, будут после этого помечены символом –. Далее, клавишей табулятора или стрелками управления курсора следует последовательно переходить на клавишу, указанного символа не имеющую, и нажимать Пробел, а затем — нераспознаваемую клавишу.

Так, например, если в каком-либо эмуляторе терминала MC по умолчанию не обрабатывает корректно нажатия функциональных клавиш с F1 по F5 и некоторых других, мы сначала переходим на клавишу F1, нажимаем клавишу Пробел и вслед за ней — клавишу F1. По исчезновении предупреждающего сообщения снова повторяем нажатие на F1 и по появившейся на ней отметке — убеждаемся в ее нормальном функционировании (см. рис. 8.24).

Таким образом, MC — быстрый и простой в освоении и использовании универсальный файловый менеджер, обеспечивающий выполнение полного комплекса файловых операций, доступ к данным на удаленных компьютерах локальной сети и выступающий также в качестве ftp-клиента.

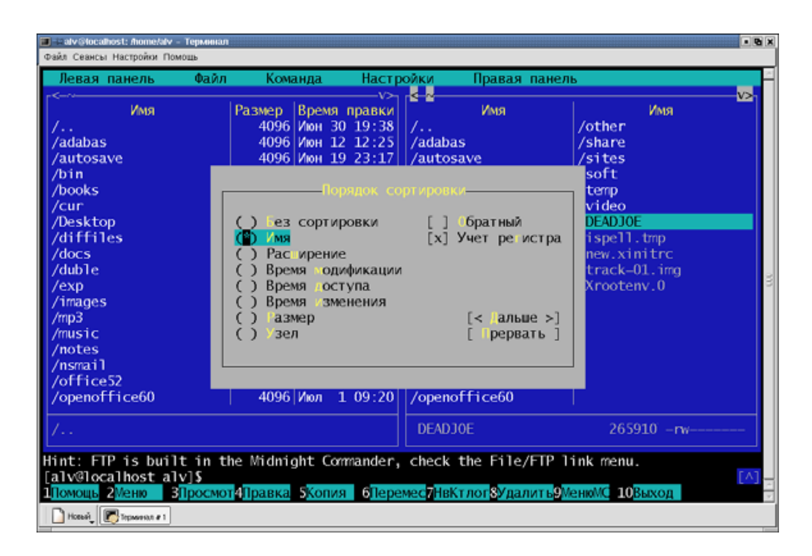


Рис. 8.10: Определение порядка сортировки

8.2 Средства архивации и компрессии

Средства архивации и компрессии необходимы как для транспортировки данных, так и для их резервного копирования. Программы-архиваторы обеспечивают сборку группы файлов или каталогов в форму, подходящую для записи на резервный носитель и последующего восстановления с него в первозданном виде. Средства же компрессии предназначены для уменьшения объема, занимаемого файлами на диске (или ином носителе).

В ОС MS DOS и Windows программы-архиваторы выступают одновременно и в роли компрессоров, но в Linux это разные операции и они выполняются разными программами.

Основным инструментом архивации в Linux является команда tar. Изначально она предназначалась для создания архивов на магнитной ленте, откуда и название (`tar` — от `Tape ARchive`), однако ныне используется просто для объединения нескольких файлов или каталогов в один архивный файл (`*.tar`) с сохранением путей, прав доступа и принадлежности. Она же позволяет развернуть архив в группу файлов или каталогов.

Формат команды tar следующий:

```
tar fo file1 file2... file99
```

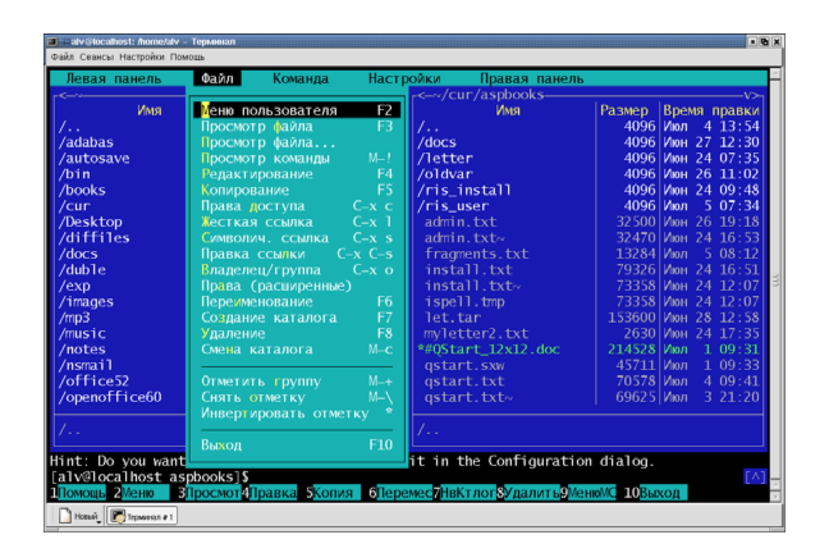


Рис. 8.11: Меню Файл — основные файловые операции

где f — параметр, определяющий выполняемую функцию, o - опции этой функции, `file*` — имена файлов для архивирования или архивов для развертывания. Для параметров и опций используется два вида нотации — сокращенная (однобуквенная) и полная. Особенностью синтаксиса tar является то, что при сокращенной нотации знак дефиса перед параметром и опциями не обязателен (хотя и допустима конструкция `tar -fo file`), а сами они пробелами не разделяются. При полной нотации перед параметром и опцией ставится двойной дефис и разделяющий их пробел:

```
tar --function --option file
```

Некоторые параметры можно задать только в полной нотации. В частности, получить справку по tar можно только командой

```
tar --help
```

но не

```
tar h или tar -h
```

В соответствии с назначением программы, в tar имеется два главных параметра — c (или `--create`) для создания нового архива, и x (или `--extract`) —

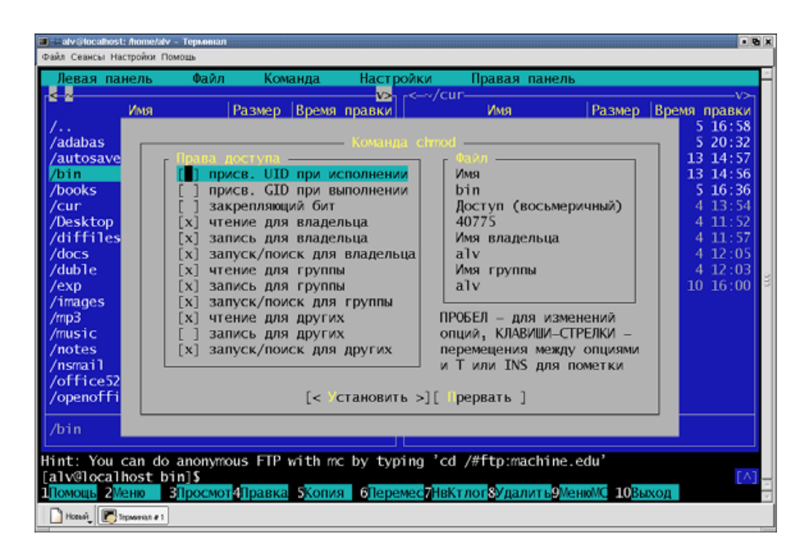


Рис. 8.12: Установление прав доступа

для развертывания существующего. Кроме того, над архивами возможны следующие действия:

- t (--list) — вывод списка файлов архива;
- d (--diff) — сравнение архива с каталогом для поиска различий;
- r (--append) — присоединение файлов к концу архива;
- u (--update) — обновление архива, то есть присоединение отсутствующих в нем файлов и замена имеющихся их более новыми версиями (если они имеются);
- A (--catenate) — присоединение одного tar-архива к другому;
- --delete — удаление файлов из архива.

Каждый параметр может иметь более чем одну опцию. Однако последней принято ставить опцию f или --file (для некоторых версий tar это является обязательным требованием), за которой следуют аргументы — имя файла создаваемого архива, а потом имена файлов для архивирования. Если опция f не указана, программа пытается создать архив (или считать его) на устройстве по умолчанию — стримере /dev/rmt0, а поскольку такового в системе, скорее всего, нет, последует сообщение об ошибке.

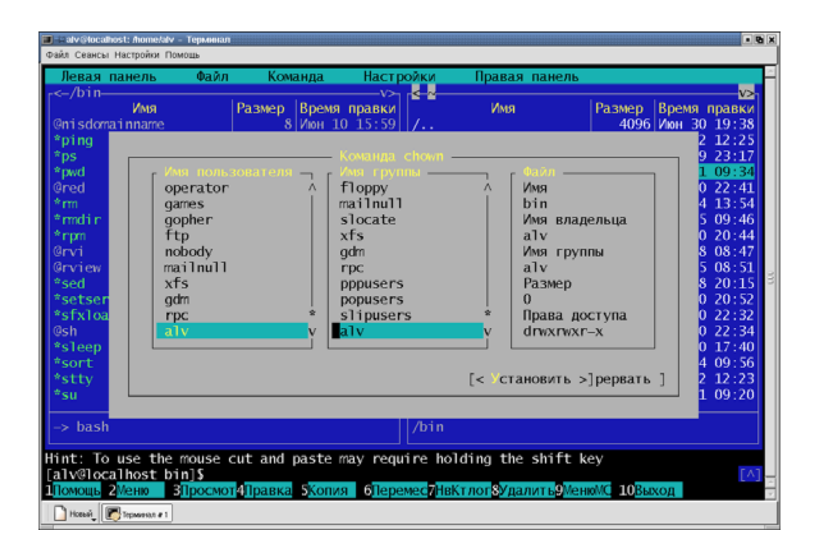


Рис. 8.13: Определение принадлежности файлов

В промежутке между параметром (например, c или x) и опцией f могут быть использованы дополнительные опции, такие как:

- v — вывод имен архивируемых/разархивируемых файлов; двойная опция vv выводит полную информацию о файлах;
- W — верификация архива;
- k — запрет на переписывание существующих файлов при разархивации;
- другие. Из них важны опции z и j, предписывающие сжимать архивируемые файлы с помощью программ компрессии gzip и bzip2 соответственно. Или, напротив, распаковывать сжатые tar-архивы при разархивации (или при просмотре архива), о чем будет сказано ниже.

С помощью tar можно архивировать не только отдельные файлы, но и целые каталоги, в том числе с подкаталогами любого уровня вложенности. Так, командой

```
tar cvf alldata.tar ~/alldata
```

каталог alldata в домашнем каталоге пользователя будет целиком упакован в архив alldata.tar, который будет помещен в текущий каталог. Если просмотреть его потом командой

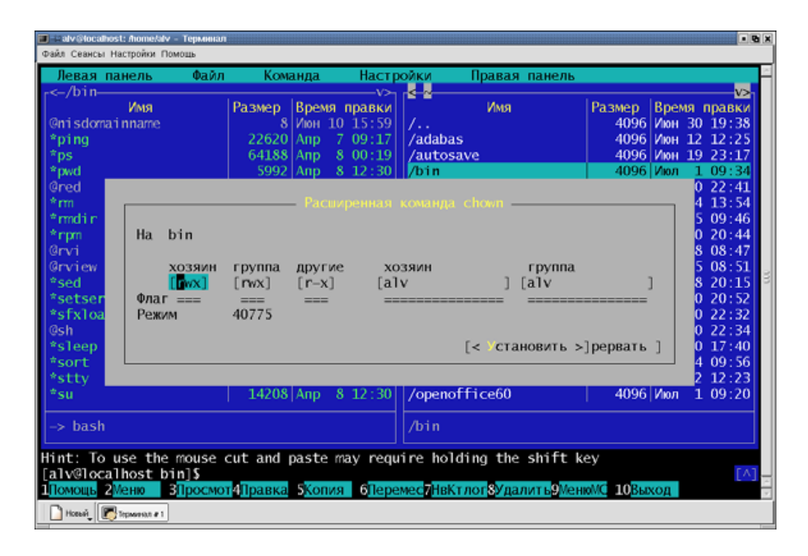


Рис. 8.14: Установка прав доступа в символьной форме

```
tar tf alldata.tar
```

можно видеть, что путь каждого файла будет включать в качестве первого элемента домашний каталог пользователя:

```
home/username/alldata
home/username/alldata/data1
home/username/alldata/data1/file1
```

и т.д. Та же команда в форме

```
tar cvf alldata.tar alldata
```

(в предположении, что текущим является домашний каталог пользователя) приведет к сохранению в архиве путей вида

```
alldata
alldata/data1
alldata/data1/file1
```

Наконец, если архивировать тот же каталог командой

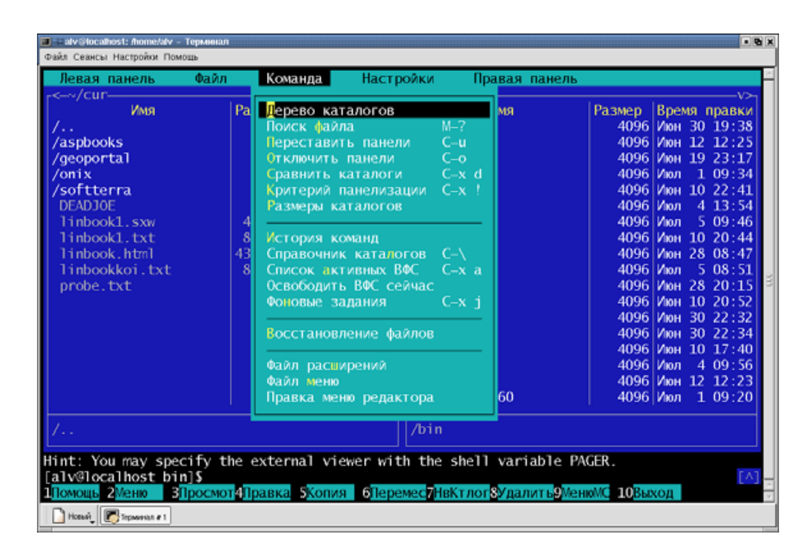


Рис. 8.15: Главное меню: пункт Команды

```
tar cvf alldata.tar alldata/*
```

пути файлов в архиве примут вид

```
alldata/data1
alldata/data1/file1
```

и т.д. Все это должно учитываться при разархивации. Обычно принята вторая форма архивирования, то есть

```
tar cvf alldata.tar alldata
```

В этом случае при его разархивации в произвольном месте файловой системы целиком командой

```
tar xvf alldata.tar
```

просто создается новый каталог `~/alldata`, куда и помещаются все распакованные файлы. Это предотвращает их смешение с существующими файлами и уничтожение последних при случайном совпадении имен.

Из созданного архива можно извлекать и отдельные файлы, что требует указания не только их имен, но и полных путей в той форме, в которой они сохранены в архиве, например:

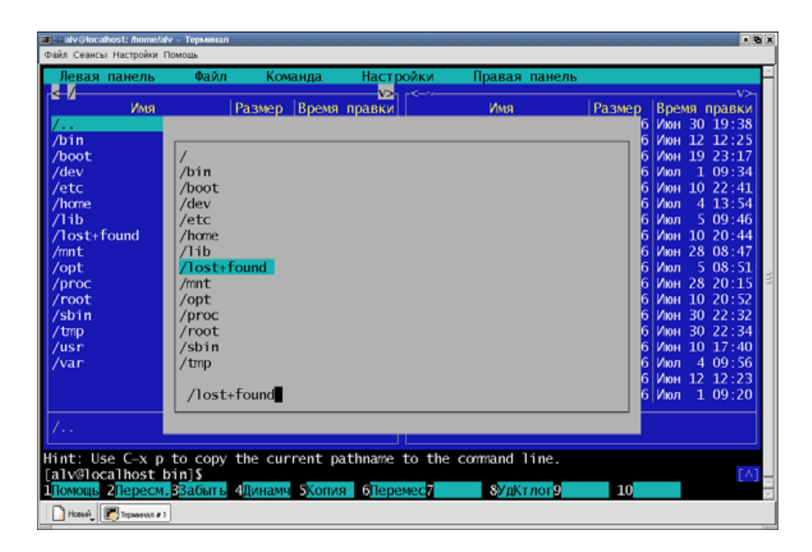


Рис. 8.16: Вывод дерева корневого каталога

```
tar xvf alldata.tar alldata/data1/file1
```

Пути эти можно предварительно узнать с помощью команды

```
tar tvf alldata.tar
```

Наиболее употребимая в Linux программа компрессии — `gzip`, предназначенная для сжатия файлов; парной ей является утилита `gunzip`, служащая для их распаковки. Впрочем, обе они, при использовании соответствующей опции, выполняют и обратную исходной операцию.

Программа `gzip` — это именно и только чистый компрессор, применимый лишь к единичному файлу. При этом (внимание!) исходный несжатый файл подменяется его сжатой копией, которой автоматически присваивается расширение `*.gz`.

Формат команды:

```
gzip file
```

в результате чего образуется файл `file.gz`, а исходный `file` исчезает.

Обратная процедура выполняется командой

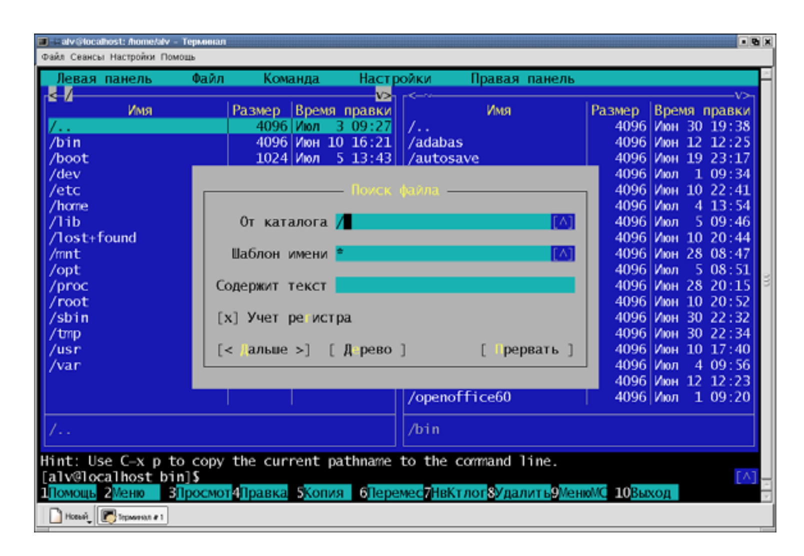


Рис. 8.17: Поиск файла

```
gunzip file.gz
```

что влечет за собой замену файла `file.gz` файлом `file`. Распаковку сжатого файла можно выполнить и командой `gzip`, для чего она дается в следующей форме

```
gzip -d file.gz
```

или

```
gzip --decompress file.gz
```

Команда `gzip` имеет и другие опции, указываемые в краткой (однобуквенной) или полной нотации. В отличие от `tar`, знак дефиса (или, соответственно, двойного дефиса) обязателен в обоих случаях.

Так, опциями `-1` ... `-9` можно задать степень сжатия и, соответственно, время процедуры: `-1` соответствует минимальному, но быстрому сжатию, `-9` — максимальному, но медленному. По умолчанию в команде `gzip` используется опция `-6`, обеспечивающая компромисс между скоростью и степенью сжатия.

Как уже говорилось, команда `gzip` (как и `gunzip`) применима только к единичному файлу. Однако есть возможность с помощью одной команды сжать

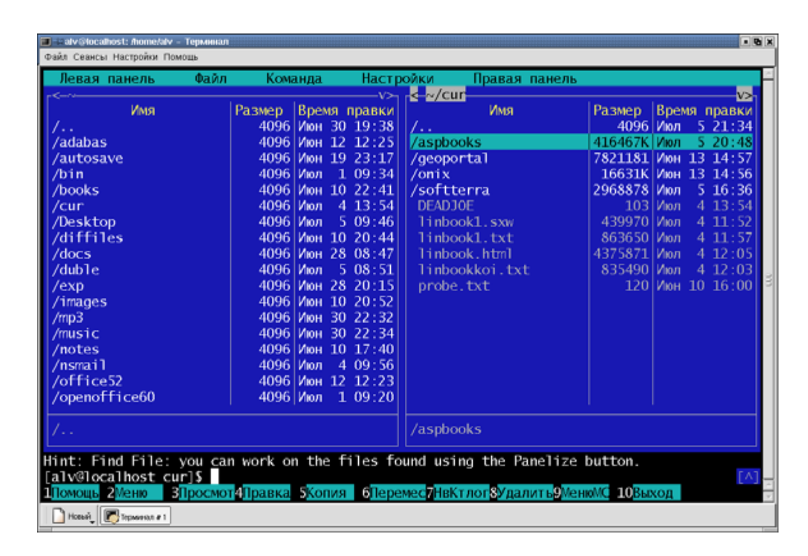


Рис. 8.18: Размеры каталогов: в правой панели — суммарный объем файлов, в левой — объем, занимаемый файлом каталога.

(или распаковать) сразу несколько файлов. Это делается двояко. Либо в качестве аргументов команды просто перечисляются подлежащие сжатию файлы

```
gzip file1 file2 ... file99
```

результатом чего будет появление в текущем каталоге файлов `file1.gz`, `file2.gz` и т.д.

Либо, если сжимаемые файлы, расположены в одном каталоге, можно использовать опцию рекурсии `-r`. Так, если требуется упаковать все файлы в каталоге `dir`, содержащем файлы `file1` и `file2` и подкаталог `subdir` с файлами `file3` и `file4`, команда дается в следующей форме:

```
gzip -r dir
```

после чего просмотр командой

```
ls dir/*
```

выдаст следующий результат:

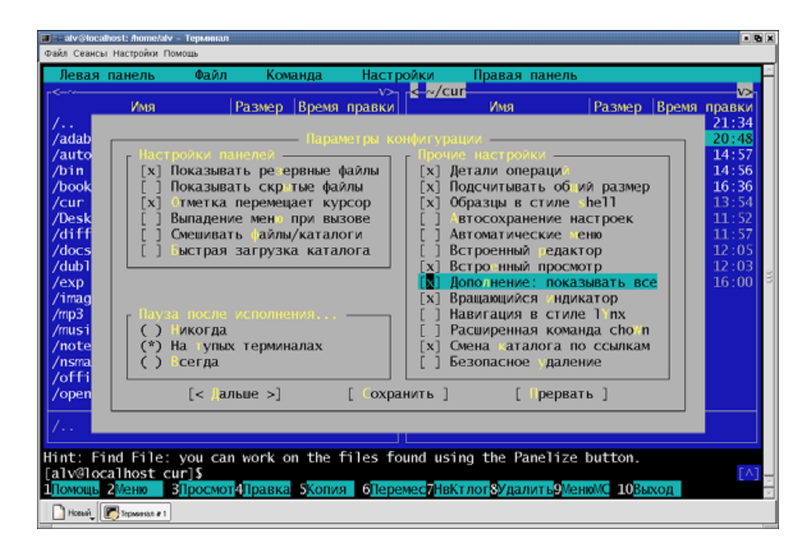


Рис. 8.19: Общие настройки MC

```
dir/file1.gz dir/file2.gz
dir/subdir:
file3.gz  file4.gz
```

С помощью команд

```
gunzip -r dir
```

или

```
gzip -dr dir
```

сжатые файлы в каталоге dir будут распакованы.

Поскольку программы `tar` и `gzip` обеспечивают каждая свою сторону создания архивов, они обычно используются совместно. Сделать это можно двумя путями. Во-первых, можно ограничиться командой `tar` с опцией `z`, например, в следующем виде

```
tar cvzf dir.tar.gz dir/
```

Обратите внимание, что расширение `*.gz` в этом случае нужно указывать в явном виде, автоматически оно к имени архива не присоединяется, и компрессированный архив будет иметь вид `dir.tar`. Поскольку в Linux расширения

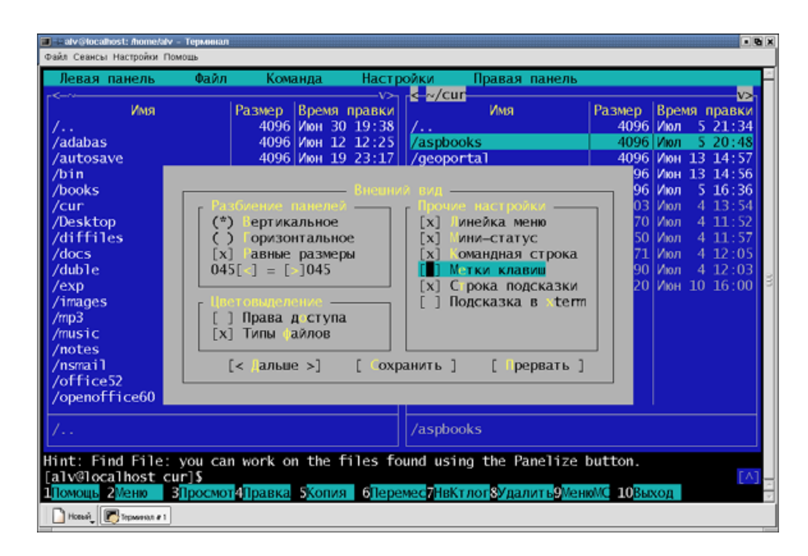


Рис. 8.20: Настройка внешнего вида MC

имен файлов не играют той роли, что в MS DOS (о чем подробнее сказано в руководстве администратора), это не помешает распаковке такого файла командой

```
tar xvzf dir.tar
```

Следует, однако, помнить, что архив сжатых файлов не может быть обновлен командой `tar` с параметрами `r` или `u`.

Другой способ — создание архивного файла командой вида

```
tar cvf dir.tar dir/
```

а затем его сжатие как единого целого командой

```
gzip dir.tar
```

В результате образуется такой же файл `dir.tar.gz`. Однако в принципе архив сжатых файлов и сжатый архивный файл — это разные вещи. Очевидно, что сжатый архивный файл не может быть ни пополнен, ни обновлен средствами `tar`.

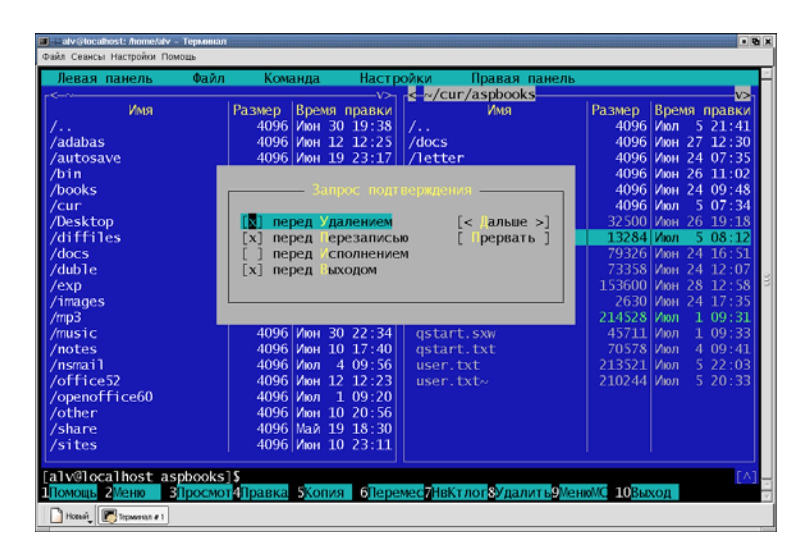


Рис. 8.21: Настройка подтверждений

Однако такая процедура возможна с помощью иных архиваторов (например, `ark`), поэтому такой формат препочтительнее архива сжатых файлов.

Компрессированные архивы, созданные сочетанием программ `tar` и `gzip` — общепринятый в UNIX-системах метод распространения файлов. В итоге почти все программы для Linux, которые можно обнаружить в Интернете, имеют формат `*.tar.gz`, а некоторые — исключительно в нем. Часто для совместимости с ОС, не допускающими двух точек в имени файла, компрессированным tar-архивам на ftp-серверах присваивается расширение `*.tgz`.

В коллекциях программ на ftp-серверах не следует путать простые архивы `*.tgz` (то есть просто переименованные файлы `tar.gz`) с бинарными пакетами того же вида, используемыми во многих дистрибутивах Linux (например, в Slackware) или в FreeBSD и OpenBSD. Последние также представляют собой упакованные tar-архивы, но включают, помимо откомпилированных собственно программных файлов, также дополнительные инсталляционные сценарии, ориентированные на конкретные дистрибутивы или ОС. Попытка напрямую установить их в **ASPLinux**, скорее всего, закончится неудачей.

Однако `gzip` — не единственная программа компрессии для Linux. В последнее время широкое распространение получил компрессор `bzip2`, обеспечивающий большую (на 10-15%) степень сжатия, хотя и менее быстродействующий.

Использование его практически идентично `gzip`, с деталями его можно озна-

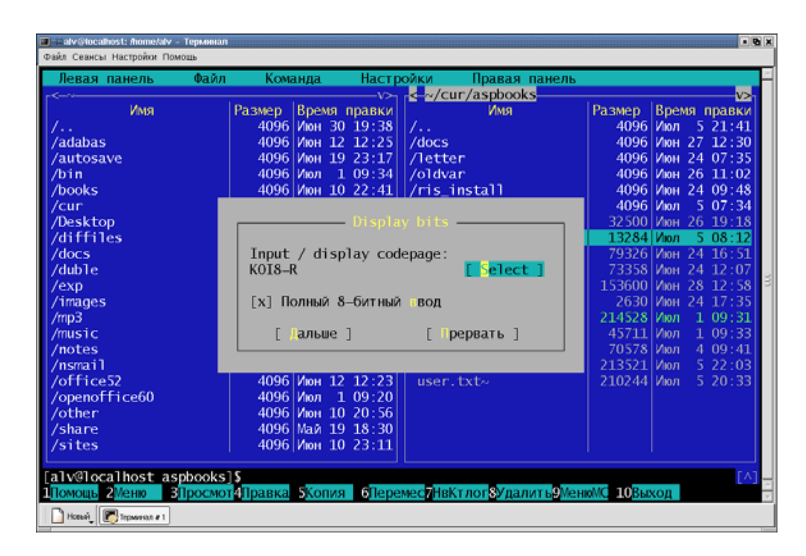


Рис. 8.22: Настройка ввода/вывода нелатинских символов

комиться с помощью `man bzip2` или `info bzip2`. Итоговый компрессированный файл получает имя вида `*.bz2` и может быть распакован командой `bunzip2`.

Следует только помнить, что форматы `*.gz` и `*.bz2` несовместимы между собой: первый не может быть распакован программой `bunzip2`, а второй — программой `gunzip`.

Тем не менее, существуют программы, способные одновременно работать с архивами форматов `tar.gz` и `tar.bz2`. К ним принадлежит **Ark**, входящий в состав *KDE* и имеющий, к тому же, удобный графический интерфейс.

Программа **Ark** представляет собой оболочку, интегрирующую в себе различные архиваторы и компрессоры командной строки. Она вызывается из стартового меню (*«Утилиты»- «Архиватор»*) или набором одноименной команды в окне терминала или строке минитерминала. После этого открывается пустое окно программы со строкой меню и панелью инструментов вверху и статусной строкой внизу (рис. 8.25).

Сразу после запуска доступно два действия — создание нового архива или открытие существующего. Первое выполняется кнопкой в панели инструментов или через меню *«Файл»*Новый. После этого предлагается дать имя архивному файлу. Формат архива распознается автоматически по расширению, поэтому оно должно быть задано в явном виде.

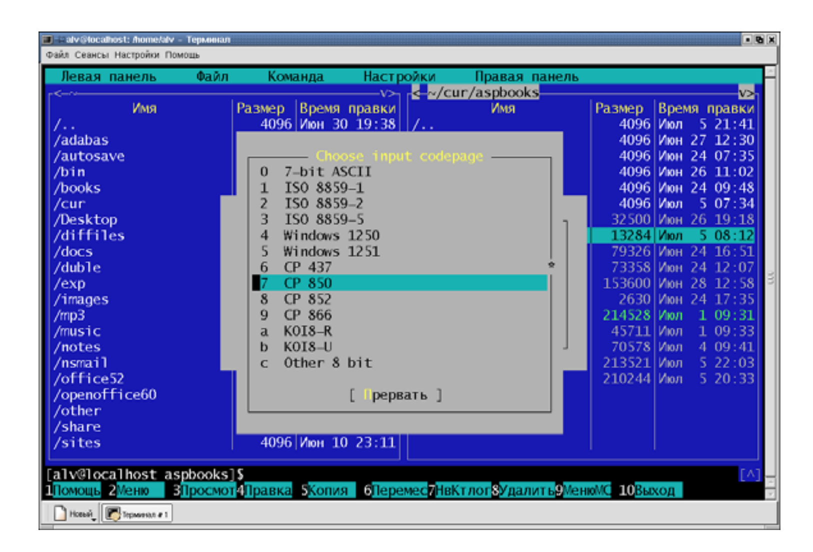


Рис. 8.23: Установка кодировок для имен файлов, передаваемых национальными наборами символов

Список поддерживаемых форматов можно посмотреть, открыв выпадающее меню поля *«Фильтр»* в панели выбора файлов (рис. 8.26). Он включает все традиционные для UNIX-систем форматы архивов, в том числе и компрессированных — `tar`, `tar.gz`, `tar.bz2`, `tar.Z`, компрессоры `gzip` и `bzip2`, а также некоторые форматы архиваторов/компрессоров MS DOS/Windows — ZIP, LHA и даже RAR.

Поскольку **Ark** — не более чем оболочка для программ-архиваторов и компрессоров, для работы со всеми теоретически доступными пакетами должны быть установлены соответствующие пакеты, то есть Linux-версии `zip` и `unzip`, `rar` и `unrar` и т.д. Большая их часть входит в состав дистрибутива **ASPLinux**, хотя некоторые придется доустанавливать из сторонних источников. Так, в дистрибутиве имеется пакет unrar для распаковки RAR-архивов, но парной утилиты `rar` для их создания не имеется.

Как уже говорилось, формат создаваемого архива определяется расширением его имени. Соответственно, если в поле выбора задать `archive.tar`, `archive.tar.gz`, `archive.tar.bz2` или `archive.zip`, выбранные позднее файлы будут упакованы соответствующим образом.

После задания имени становятся доступными добавление файла или каталога через меню (*«Действие»- «Добавить файл»* или *«Добавить каталог»*) или одноименными кнопками на панели инструментов. Это автоматически вызыва-

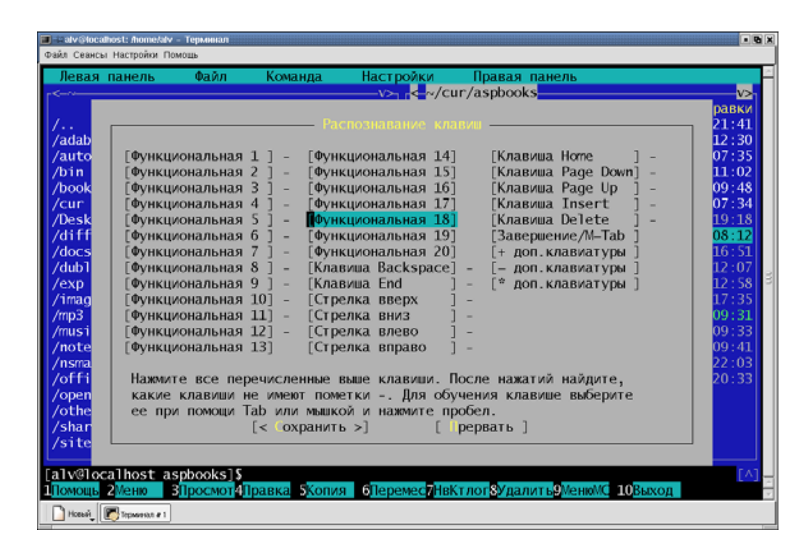


Рис. 8.24: Распознавание клавиш

ет начало процесса архивации и (или) компрессии. Никакой индикации процесса, правда, не выводится, лишь по завершении в окне программы появляется список входящих в архив файлов (рис. 8.27).

Такой же список выводится в окне и в том случае, если вместо создания нового архивного файла был открыт уже существующий. После этого активизируются пункты в меню *«Правка»* и *«Действия»*. С помощью первого меню (рис. 8.28) можно выбрать один или несколько файлов (введя их имена вручную), выбрать все файлы, снять или обратить выделение, а также просмотреть вывод архиватора (рис. 8.29).

Над выбранными файлами через меню *«Действия»* становятся доступны

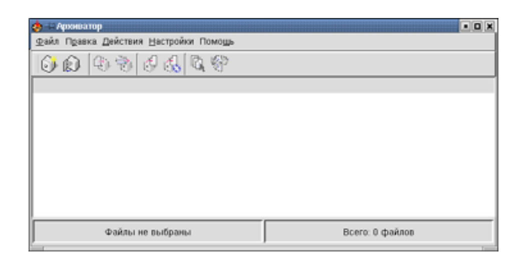


Рис. 8.25: Архиватор **Ark** после запуска

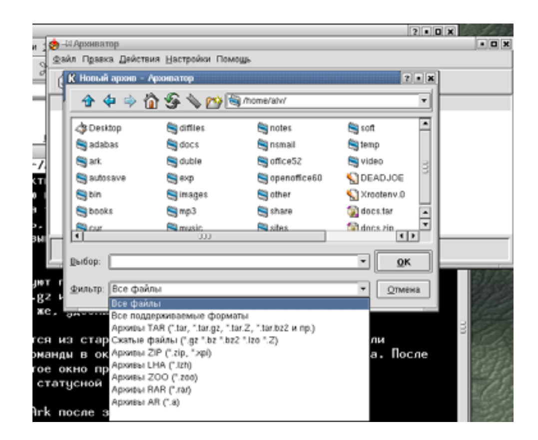


Рис. 8.26: Создание архива в **Ark**

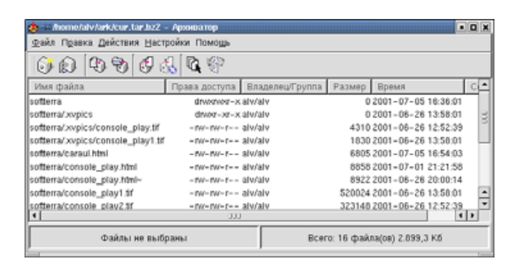


Рис. 8.27: Список файлов архива после его создания или открытия

следующие операции (рис. 8.30): *«Удалить»*, *«Распаковать»*, *«Вид»*, *«Открыть»*, *«Редактировать»*.

Смысл первых двух пунктов понятен, остальные три вызывают панель выбора приложения для просмотра, открытия или редактирования выбранного файла.

Кроме того, разумеется, к открытому архиву можно в любой момент добавить файл или каталог.

Для конфигурирования **Ark** предназначено меню *«Настройки»* (рис. 8.31). Здесь можно включить/выключить показ панели инструментов и строки состояния, скорректировать состав панели (в дополнение к имеющимся по умолчанию кнопкам добавить любые другие, соответствующие действиям из пунктов главного меню), определить горячие клавиши.

Однако главное для пользователя — настройки каталогов, откуда по умолчанию добавляются файлы в существующие архивы, куда записываются вновь

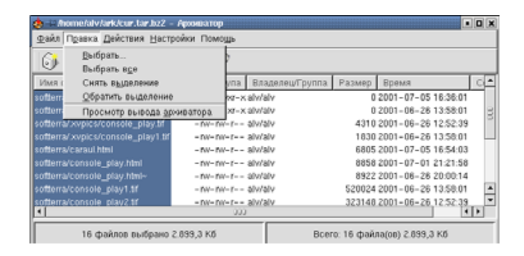


Рис. 8.28: Главное меню **Ark**, пункт *«Правка»*

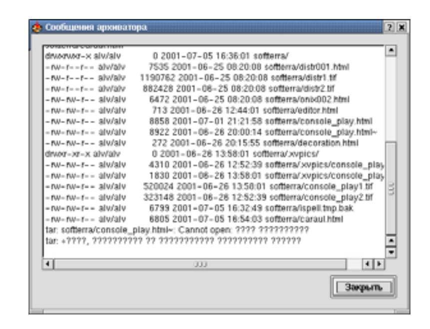


Рис. 8.29: Меню *«Правка»*, просмотр вывода архиватора

созданные архивы и т.д. (рис. 8.32).

Иными словами, программа **Ark** упрощает создание архивов, дополняя стандартные утилиты такими возможностями, как интерактивное пополнение и обновление каталогов, удаление из них ненужных файлов и т.д. Однако она не предоставляет возможности ряда тонких настроек, например, вариаций степени/скорости сжатия, и потому не заменяет утилит командной строки для архивации и компрессии.

8.3 Средства резервного копирования

Традиционным мобильным носителем информации в Linux выступает магнитная лента. Однако это не лучшее решение для конечного пользователя. С другой стороны, накопители типа Zip (и тем более дискеты) не адекватны современным объемам жестких дисков и имеют чрезмерно высокую удельную (на единицу объема информации) стоимость. Поэтому настоящий раздел будет посвящен средствам записи CD-R/RW, устройства для работы с которыми (особенно относительно дешевые, с интерфейсом ATAPI) получают все

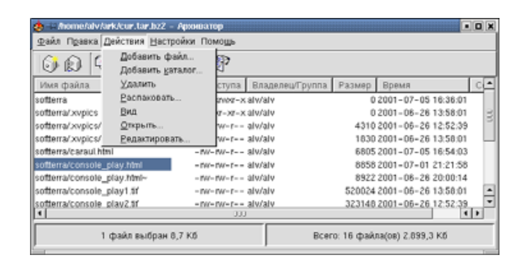


Рис. 8.30: Меню *«Действия»*

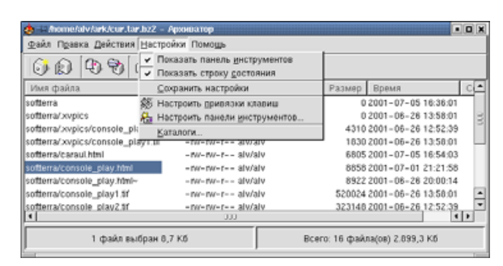


Рис. 8.31: Меню *«Настройки»*

большее распространение в настольных системах.

К сожалению, именно работа с ATAPI CD-R/RW до сих пор остается слабой стороной Linux. Если записывающие CD-устройства с интерфейсом SCSI, как правило, не вызывают никаких проблем (если системой поддерживается данный SCSI-адаптер), то отлаженной прямой поддержки ATAPI CD-R/RW в Linux пока что нет.

Тем не менее использование таких устройств вполне возможно. Оно осуществляется путем эмуляции SCSI-протокола через IDE-интерфейс. Для этого требуется компиляция ядра с соответствующими опциями, что в **ASPLinux** выполняется по умолчанию при установке системы. Далее следует лишь подгрузить соответствующий модуль, что делается от лица администратора командой

```
modprobe ide-scsi
```

ответом на что должно быть сообщение типа

```
scsi0 : SCSI host adapter emulation for IDE ATAPI devices
scsi : 1 host.
Type: Direct-Access  ANSI SCSI revision: 00
Detected scsi removable disk sda at scsi0, channel 0, id 0, lun 0
```

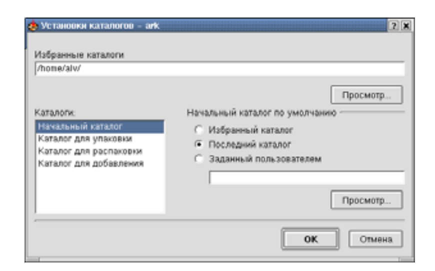



Рис. 8.32: Меню *«Настройки»*, выбор каталогов для архивации

```
Vendor: MATSHITA  Model: CD-RW  CW-7586 Rev: 1.00
Type: CD-ROM  ANSI SCSI revision: 02
Detected scsi CD-ROM sr0 at scsi0, channel 0, id 1, lun 0
sr0: scsi3-mmc drive: 32x/32x writer cd/rw xa/form2 cdda tray
sda : status = 0, message = 00, host = 0, driver = 28
sda : extended sense code = 2
sda : block size assumed to be 512 bytes, disk size 1GB
```

Окончательно правильность загрузки модуля `ide-scsi` проверяется утилитой для записи CD-R/RW — `cdrecord`. Подробнее о ней будет сказано ниже, пока же достаточно запустить ее с опцией тестирования SCSI-устройств

```
cdrecord -scanbus
```

после чего будет выведено сообщение об обнаруженных SCSI-адаптерах:

```
Cdrecord 1.9 (i686-pc-linux-gnu) Copyright (C) 1995-2000 Jrg Schilling
\Linux{} sg driver version: 2.1.39
Using libscg version 'schily-0.1'
scsibus0:
0,0,0     0)'MATSHITA' 'CD-RW  CW-7586  ' '1.00' Removable CD-ROM
0,1,0     1) *
0,2,0     2) *
0,3,0     3) *
0,4,0     4) *
0,5,0     5) *
0,6,0     6) *
0,7,0     7) *
```

Детали сообщений и в первом, и во втором случае могут различаться в зависимости от других имеющихся устройств (читающих CD-ROM и Zip с IDE-интерфейсом), которые также иногда могут определяться как эмулирующие SCSI. Однако если CD-R/RW будет в списке SCSI-устройств, проблем с его использованием быть не должно.

Собственно для записи дисков CD-R/RW в Linux применяется пара утилит - mkisofs и cdrecord. Первая создает т.н. образ диска (iso-image), вторая обеспечивает его запись на CD. Все остальные программы этого назначения представляют собой их интегрирующие оболочки.

Утилита mkisofs способна создавать образы диска в чистом формате ISO9660, воспринимаемом на всех платформах (включая MS DOS), в формате ISO9660 с т.н. расширением Joliet, разработанном для Windows 9x и способном передавать длинные имена файлов, в том числе и содержащие национальные символы, а также с расширением Rock Ridge, используемом на UNIX-платформах для передачи их специфических типов файлов (символических ссылок, например) и атрибутов (прав доступа и принадлежности). Доступно и расширение HFS для использования в MacOS. Для любой из этих файловых систем возможно создание загрузочных дисков.

Вследствие своей универсальности mkisofs содержит очень большое количество опций, с которыми можно ознакомиться через man mkisofs и info mkisofs. Более краткую, но также перегруженную подробностями справку можно получить посредством

```
mkisofs --help
```

С помощью утилиты cdrecord созданный образ того или иного формата записывается на носитель CD-R/RW. Эта программа также изобилует опциями, ознакомиться с которыми можно на страницах экранной документации.

Приведем типовой рецепт подготовки образа и записи CD-R/RW, предназначенных для платформы PC и доступных Linux и Windows 9x. Образ диска создается командой следующего вида:

```
mkisofs -r -J -o cd_image directory/
```

где -o — опция, предписывающая записать образ в файл, cd_image — имя этого файла, а directory/ — каталог с данными, подлежащими записи. Опция -r предписывает применить расширение Rock Ridge. Опция -J — расширение Joilet.

Далее созданный образ можно протестировать, воспользовавшись свойством Linux монтировать файлы как разделы диска:

```
mount -t iso9660 -o loop cd_image /mnt/cdrom
```

и просмотреть его в каталоге /mnt/cdrom, как если бы там был смонтирован уже записанный реальный диск. После чего образ следует размонтировать командой



Рис. 8.33: **X-CD-Roast** — начальное меню

```
umount /mnt/cdrom
```

Затем командой вида

```
cdrecord -v dev=0,0,0  -data  cd_image
```

осуществляется собственно запись. В этой команде значение dev=0,0,0 должно соответствовать выводу команды cdrecord -scanbus для этого устройства.

Использование утилит mkisofs и cdrecord имеет много тонкостей. И потому, возможно, проще будет воспользоваться какой-либо из программ-оболочек графического режима. Одна из наиболее развитых программ такого рода — **X-CD-Roast** — входит в состав дистрибутива **ASPLinux**.

Работа с программой **X-CD-Roast** возможна исключительно в режиме администратора. Запускается она командой xcdroast в окне терминала или строке минитерминала, вызывающей появление заставки программы с кнопками: **«SETUP»**, **«DUPLICATE CD»**, **«CREATE CD»**, **EXIT** (рис. 8.33).

Прежде любых других действий программу **X-CD-Roast** необходимо настроить. Вход в меню настройки — через кнопку **«SETUP»** начального меню. После этого на панель выводится список распознанных SCSI-устройств (или устройств, эмулирующих SCSI), а остальные настройки осуществляются на четырех следующих закладках (рис. 8.34).

Переходя на соответствующие закладки, можно настроить:

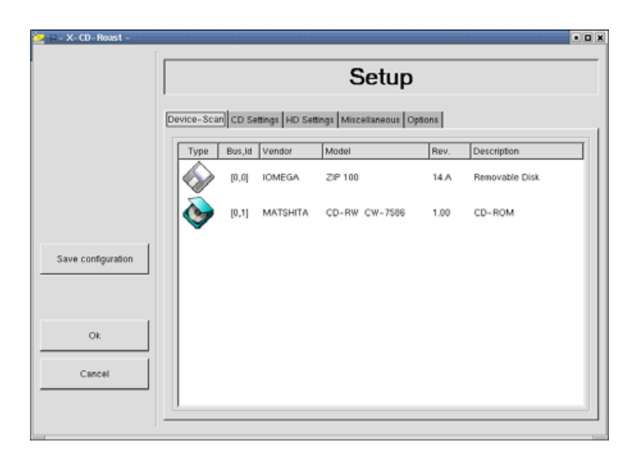


Рис. 8.34: Настройка **X-CD-Roast** — список SCSI-устройств

- тип и скорость записывающего устройства; первый обычно правильно опознается программой, но скорость лучше указать явным образом;
- здесь же определяется тип, режим и скорость читающего устройства, что необходимо для прямого копирования дисков (рис. 8.35);
- раздел для помещения создаваемого образа диска и точка его монтирования (рис. 8.36). Это единственная необходимая опция настройки, без которой программа откажется работать.
- опции воспроизведения звука, сетевые настройки и т.д.

Выполнив настройки, их следует сохранить, нажав кнопку **«SAVE CONFIGURATION»**, после чего происходит возврат к начальному меню. Для создания образа диска и его последующей записи здесь следует нажать кнопку **«CREATE CD»**, вызывающую соответствующее меню (рис. 8.37).

Выбрав в левой стороне панели кнопку **«MASTER TRACKS»**, можно выбрать каталоги для записи. Для этого курсор фиксируется на нужном каталоге в их дереве (в правом окне панели) и нажимается кнопка **«ADD»** (рис. 8.38). Список выбранных каталогов выводится в левом окне.

Далее в закладке ISO9660 options определяется файловая система будущего диска. Программой **X-CD-Roast** поддерживаются следующие расширения стандартной ISO9660 (рис. 8.39):

- MS DOS — вариант ISO9660, допускающий имена файлов не длиннее шаблона 8.3.

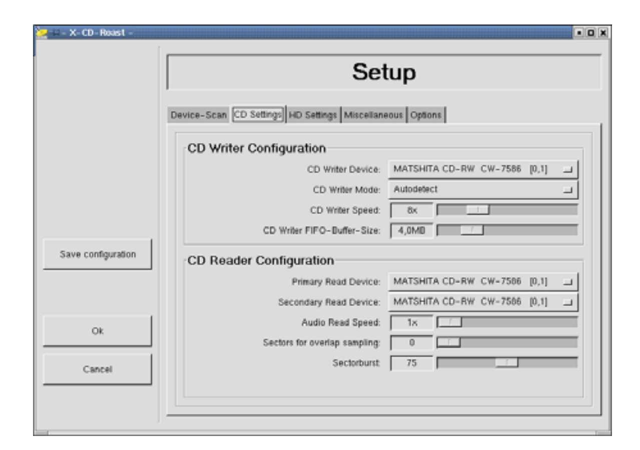


Рис. 8.35: Настройки записывающего и читающего устройств

- UNIX Rock Ridge, обеспечивающий воспроизведение таких особенностей файловой системы Linux, как длинные имена файлов, права доступа к ним, символические ссылки и т. д. Диски, записанные в этом формате, не могут быть прочитаны в Windows.
- Joliet extensions, дающая возможность воспроизведения длинных имен файлов в стиле Windows 9x. При этом, однако, теряются атрибуты файлов Linux (в частности, права доступа и принадлежности) и исчезают символические ссылки.
- Сочетание расширений Rock Ridge и Joliet, позволяющее не потерять информацию о файлах Linux, с одной стороны, и читать записанные диски под Windows — с другой.

Кроме того, все необходимые опции файловой системы (Joliet extensions, поддержка символических ссылок и прочее) можно указать вручную, отметив соответствующие переключатели; после чего сохранить эту заказную систему как используемую по умолчанию.

Далее в закладках *«Boot options»* и *«ISO9660 header»* можно при необходимости установить опции загрузки (образ, с которого будет записываться загрузочный сектор, если таковой создается) и идентификаторы CD (метка тома, аннотация и т.д.). После чего нужно перейти к закладке *«Create session/image»*.

Здесь перед пользователем две возможности: создание традиционного файла iso-образа с записью его на винчестер и запись CD «на лету» (рис. 8.40).

Для осуществления первой нужно подсчитать суммарный объем выбранных

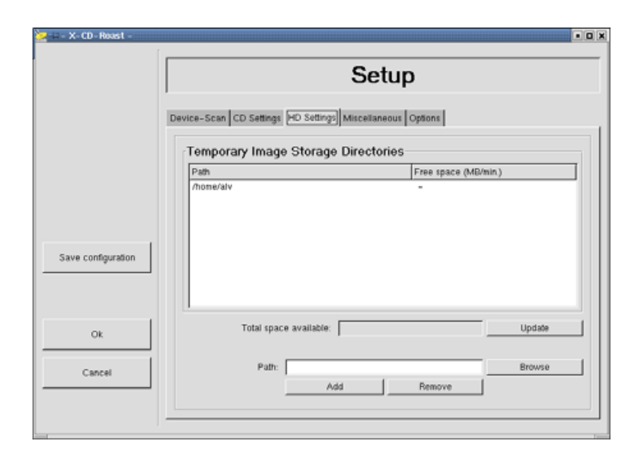


Рис. 8.36: Настройка разделов диска

данных, соответствующий размеру образа, и нажать кнопку **«MASTER TO IMAGE FILE»**, после чего начнется процесс его создания и записи в определенный настройками каталог. По завершении процесса (который может занять, в зависимости от мощности машины, от нескольких минут до нескольких десятков минут) будет выведено сообщение об этом; нажав кнопку **«OK»**, следует вернуться в меню создания диска и выбрать в нем кнопку **«WRITE TRACKS»**.

Здесь сначала, воспользовавшись закладкой *«Layout tracks»*, выбирается образ диска для записи (рис. 8.41). Для этого в правом окне панели отмечается соответствующий файл (имеющий по умолчанию вид `track-99.img`) и нажимается кнопка **«ADD»**. Результат выводится в левом окне, одновременно ниже, в поле *«Size»*, подсчитывается его размер и примерное время записи (мин: сек) при заданной в настройках скорости (на рис. 8.41 приведено время для односкоростной записи).

Нажатием кнопки **«ACCEPT TRACK LAYOUT»** подтверждается выбор и происходит возврат к закладке **«Write tracks»** (рис. 8.42). Здесь устанавливаются такие параметры, как:

- размер диска (63 минуты, 74 минуты, 80 минут) в зависимости от объема чистого диска, обычно указываемого на упаковке;
- режим записи сессии — Disk-At-Once или Track-At-Once;
- необходимость тестовой симуляции перед записью;
- выдвижение диска после записи.

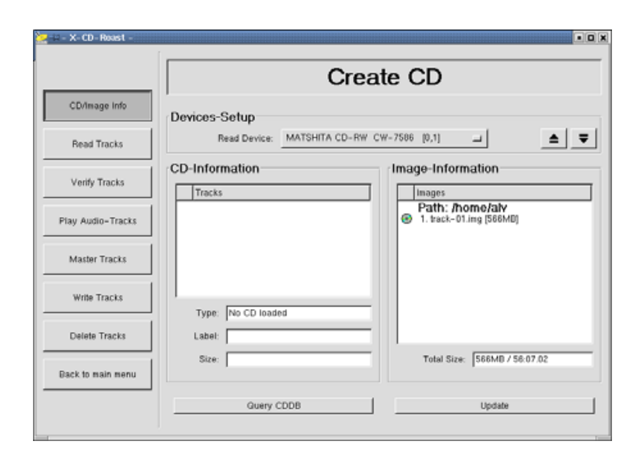


Рис. 8.37: Меню создания образа диска и его записи

После этого можно приступить к записи диска (**«WRITE TRACKS»**) или финализировать ранее записанный (в режиме Track-At-Once) диск. Для дисков CD-RW можно указать необходимость очистки ранее записанного содержания и ее режим: полная очистка диска, минимальная очистка, очистка трека или последней сессии (рис. 8.43). Очистка CD-RW может осуществляться и независимо от записи.

В текущей версии **X-CD-Roast** появилась возможность записи дисков без предварительного создания образа. Для этого нужно обратиться к правой стороне панели Create session/image (см. рис. 8.40), установить там размер диска в минутах и прочие необходимые переключатели (симуляции, выдвигания диска и т.д.) и нажать кнопку **«MASTER AND WRITE ON-THE-FLY»**. После этого начнется создание виртуального образа диска (без записи его на винчестер в виде файла) и сразу, без перехода — его запись на CD. Это быстрее, чем предварительное создание образа, но требует мощного компьютера и (или) большого кэша записывающего устройства во избежание опустошения буфера.

Еще одна программа-оболочка графического режима для записи дисков **K3b** успешно развивается в последние годы. Ее интерфейс довольно сильно напоминает интерфейс аналогичной программы **Nero**, которая работает под Windows (рис. 8.44).

Конечно же, **K3b** базируется на стандартных утилитах для записи дисков. Сюда входят такие программы, как `cdrecord`, `cdrdao`, `mkisofs` и `cdparanoia`. Поскольку **K3b** было выпущено как приложение для среды KDE3, необходимо установить соотвествующие версии библиотек. Естественно, запускать KDE для использования **K3b** не обязательно, но внутри однородной среды такая

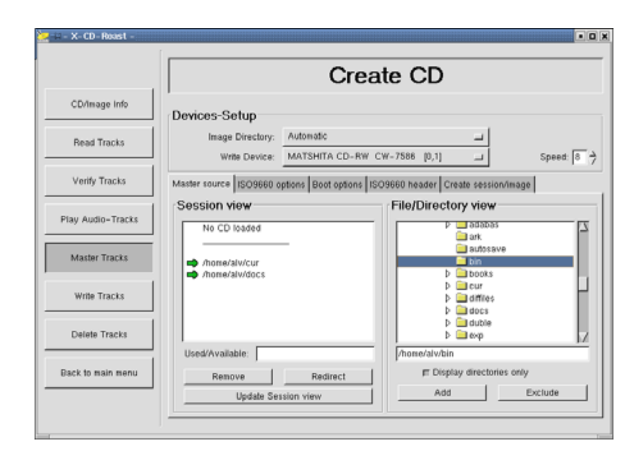


Рис. 8.38: Выбор каталогов для записи

оболочка предоставит более комфортабельные условия пользователю. Если предполагается работа с видеотреками, установите программу `transcode` (рис. 8.45).

В отличие от **X-CD-Roast** для доступа к устройству записи не нужны привелигированные права, а всего лишь занесенение пользователей, которым разрешено записывать диски в предопределенную группу.

Чтобы начать работу с программой **K3b**, просто запустите ее и следуйте за подсказками, а также обратите внимание на интуитивно-понятный интерфейс. **K3b**, к примеру, позволяет копировать CDROM и аудио CD «на лету» или вначале скопировать их на жесткий диск, а потом записать на CD, во избежание опустошения буфера. Все, что необходимо сделать — это нажать на пиктограмму *«Копировать CD»* в панели инструментов и выбрать с какого устройства считать данные и на какое затем записать их.

Также есть возможность создавать виртуальный iso-образ или иными словами записывать не образ на CD, а набор обычных файлов точно также, как это делает **Nero**. Для этого необходимо создать «новый проект» (и затем выбрать тип диска между аудио или обычным диском с данными). Затем нужно составить содержание диска, которое делается простым перетаскиванием необходимых для копирования файлов в режиме drag and drop. Внизу экрана индикатор прогресса покажет сколько свободного места на диске и сколько необходимо для записи выбранных файлов (рис. 8.46).

Однако, следует заметить, что **K3b** не просто оболочка для записи CD. С помощью нее также можно кодировать видеопотоки и даже переконвертировать DVD диски в другие форматы в целях создания резервных копий.

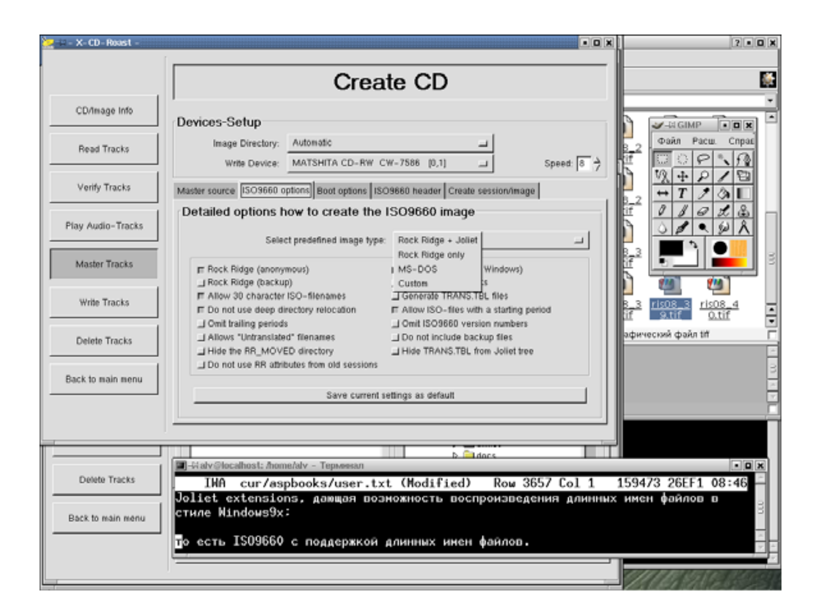


Рис. 8.39: Выбор файловой системы для образа диска

Конечно же в последнем случае необходимо помнить, что большинство DVD защищены соответствующими правами и копии будут нелегальными.

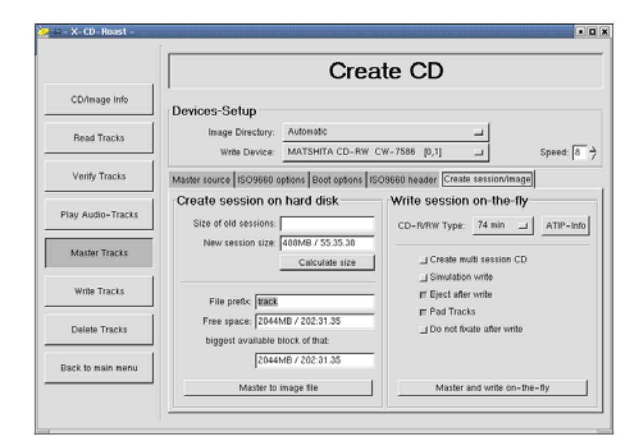


Рис. 8.40: Определение параметров сессии

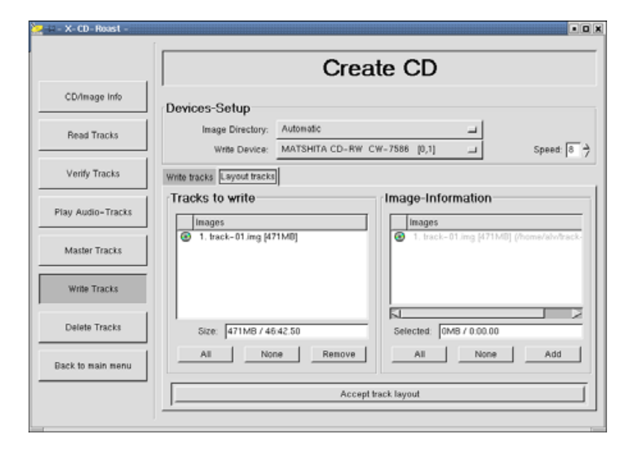


Рис. 8.41: Выбор образа диска для записи

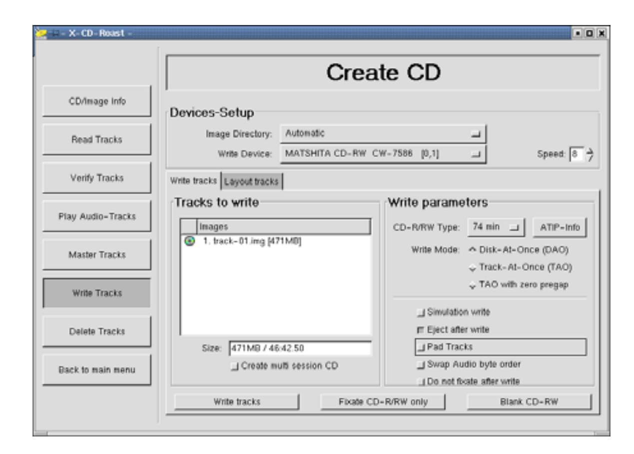


Рис. 8.42: Режимы и параметры записи

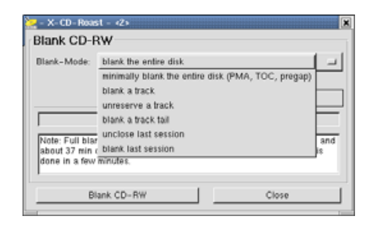


Рис. 8.43: Режимы очистки диска CD-RW

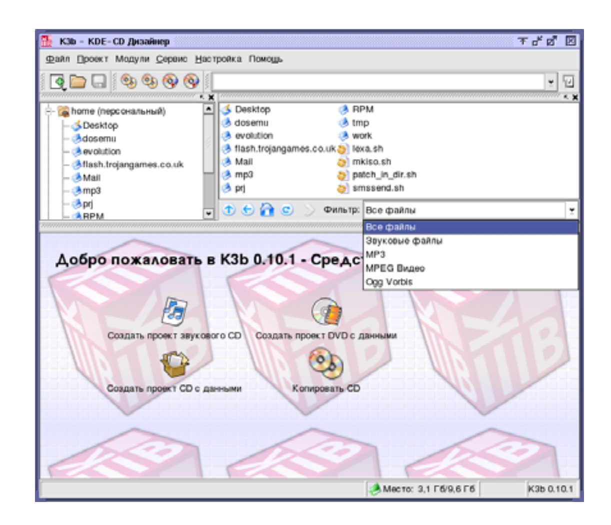


Рис. 8.44: Основное окно программы **K3b**

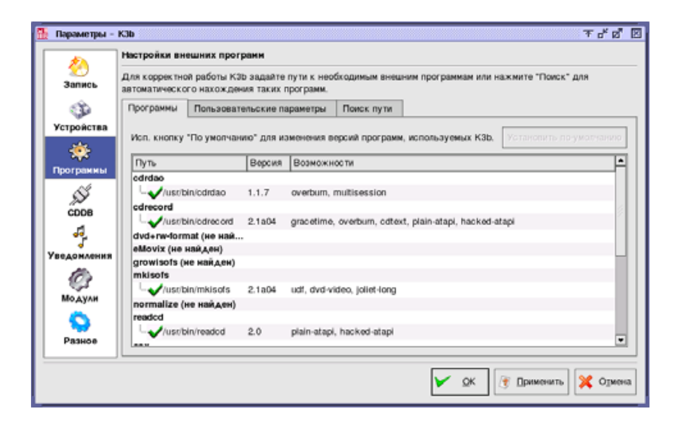


Рис. 8.45: Окно настройки параметров программы

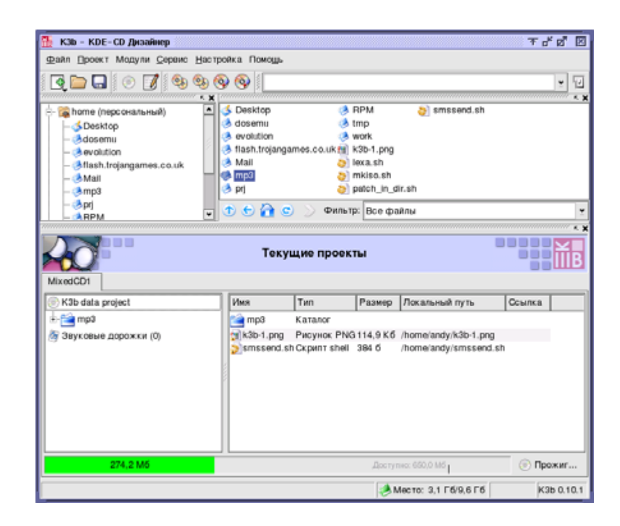


Рис. 8.46: Создание и наполнение будущего CD

Глава 9

Konqueror — файловый менеджер и браузер

Konqueror — штатный файловый менеджер интегрированной графической среды *KDE3* и является неотъемлемой ее частью. Он обеспечивает доступ к файлам на локальном компьютере, по сети и даже выполняет функции браузера. Кроме того, возможности графического интерфейса и визуального манипулирования объектами в нем интегрированы со скоростью и прозрачностью интерфейса командной строки, основанного на прямых командных директивах. Наконец, konqueror отличается почти неограниченной настраиваемостью интерфейса и функциональности.

Иными словами, **Konqueror** — универсальный инструмент для практически любых действий пользователя. Чем и обусловлено посвящение ему отдельной главы.

9.1 Общее описание

При первом запуске **Konqueror** открывается стандартное окно *KDE* со всеми его управляющими элементами (рис. 9.1). Внутри окна — строка главного меню, инструментальная панель с кнопками-пиктограммами, адресная строка, отражающая путь до локального файла или URL удаленного. Ниже — две вертикально расположенные панели. В левой — дерево каталогов, в правой — содержание текущего каталога в виде пиктограмм, сопровождаемое статусной строкой. Панели по умолчанию синхронизированы — переход по дереву каталогов вызывает автоматическое изменение содержания правой панели.

В общем, исходя из вида по умолчанию, **Konqueror** можно было бы отнести к клонам Windows Explorer. Однако интерфейс его, по желанию пользователя, может принять любой внешний облик. Так, можно скрыть дерево каталогов, а освободившееся пространство разбить на любое количество па-

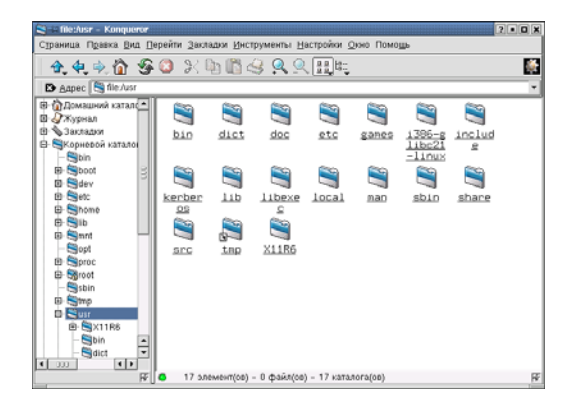


Рис. 9.1: Файловый менеджер **Konqueror** — вид по умолчанию при первом запуске

нелей, расположенных произвольным образом. Например, на две вертикально расположенные панели, подобно привычному многим интерфейсу Norton Commander.

Пиктограммы на панелях можно заменить текстовыми надписями, причем любого цвета и на любом фоне. Например, белые надписи на синем фоне способны создать зрительную иллюзию работы в Norton Commander (или Midnight Commander) — не будет доставать только командной строки.

Однако в **Konqueror** можно включить не просто командную строку, но полноценное окно эмуляции терминала — с историей команд, их дополнением посредством клавиши Tab, вставкой команд средней клавишей мыши, возможностью просмотра в любом направлении (как через полосу скроллинга, так и комбинацией клавиш Shift+PageUp/PageDown), выдачей сообщений об ошибках. Короче, всеми возможностями традиционной текстовой Linux-консоли. В итоге **Konqueror** приобретет вид, подобный показанному на рис. 9.2.

На представленном рисунке обратите внимание на переключатели внутри каждой панели (круглый слева и квадратный — справа). Первый предназначен для перехода на соответствующую панель, индицируя цветом текущую, второй — для синхронизации панелей. Так, синхронизация эмулятора терминала с одной из панелей или с деревом каталогов приводит к тому, что любые перемещения по файловой системе с помощью мыши приводят в смене текущего каталога в терминале.

Это еще далеко не все возможности модификации внешнего вида **Konqueror**, который может быть настроен в соответствии с любыми специфическими потребностями, о чем будет сказано в одном из следующих разделов.

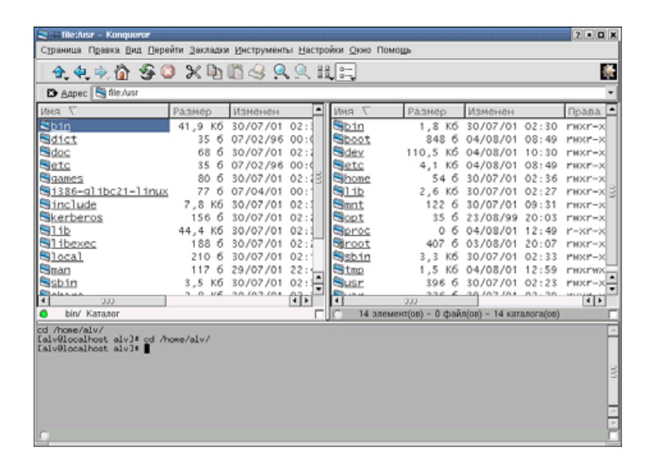


Рис. 9.2: Двухпанельное представление **Konqueror** с включенным эмулятором терминала

9.2 Основные функции

Управление файлами **Konqueror** осуществляет несколькими способами: через пункты главное меню, через контекстное меню, вызываемое щелчком правой клавиши мыши на объекте, и путем простого манипулирования мышью. Ряд опций доступен через инструментальную панель (неограниченно настраиваемую, как будет показано ниже). Многие команды меню дублируются комбинациями горячих клавиш.

Следует сразу заметить, что главное меню также является контекстно чувствительным, и содержание его пунктов зависит от текущей позиции, то есть от того, находимся ли мы в файловых панелях, в дереве каталогов или в эмуляторе терминала.

Пункты главного меню следующие:

- *«Страница»*,
- *«Редактирование»*,
- *«Вид»*,
- *«Перейти»*,
- *«Закладки»*,
- *«Инструменты»*,

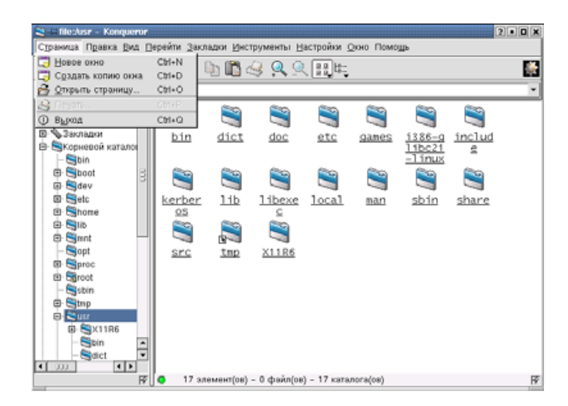


Рис. 9.3: Меню *«Страница»*

- *«Настройки»*,
- *«Окно»*,
- *«Помощь»*.

Они активизируются наведением курсора мыши или нажатием клавиши `Alt`. После чего ниспадающее меню открывается щелчком левой кнопки мыши или нажатием клавиши `Enter`. Чтобы получить представление о возможностях konqueror, рассмотрим пункты меню последовательно.

В меню *«Страница»* (рис. 9.3) можно видеть пункты *«Новое окно»* и *«Создать копию окна»* (различие между ними в том, что первый открывает окно с параметрами по умолчанию, а второй наследует таковые текущего окна), а также *«Открыть страницу»* и *«Выход»*, смысл которых очевиден.

В меню *«Правка»* (рис. 9.4) сгруппированы операции отмены, копирования, вырезания, вставки и тому подобного. Обращают на себя внимание различные режимы удаления файлов — помещения в корзину, удаления средствами операционной системы, уничтожения файла со всеми его ссылками.

Здесь же доступ к свойствам файла или каталога, включающим права доступа с возможностью их изменения в рамках текущей компетенции пользователя (или, соответственно, суперпользователя).

В этом же меню есть чрезвычайно полезный пункт — *«Создать новый»* (то есть каталог, файл, устройство или URL). Среди файлов могут быть созданы пустой файл HTML (с минимально необходимыми тэгами `title`, `html`, `body`) или `plain text`, документы в форматах офисного пакета **KOffice** (тексты, электронные таблицы, векторные рисунки, презентации), ссылки на приложения или адреса URL. Среди устройств могут быть созданы CD- или флоппи-

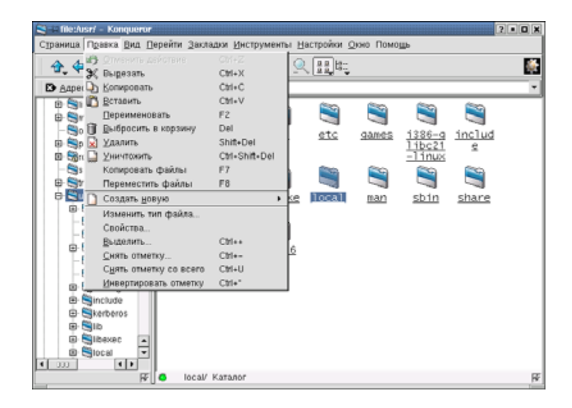


Рис. 9.4: Меню *«Правка»*

приводы — то есть в текущий каталог помещаются ссылки на соответствующие физические устройства для быстрого к ним обращения.

В меню *«Вид»* (рис. 9.5) для начала можно определить режим просмотра файлов и каталогов (с пиктограммами, в текстовом виде, многоколоночном и т.д.).

Включение пункта *«Использовать index.html»* переводит менеджер в режим браузера при попадании в каталог с индексным файлом web-сайта. Здесь же включается синхронизация панелей и показ скрытых файлов, определяются вид пиктограмм, формат списка файлов, цвет и узор фона, о чем подробнее будет идти речь в разделе о настройках. Все произведенные через меню Вид изменения сохраняют силу лишь в текущем сеансе. Наконец, через это же меню осуществляется и печать.

Название меню *«Перейти»* (рис. 9.6) отражает следующее содержание: это переход на один уровень вверх или в домашний каталог, быстрое перемещение к нескольким фиксированным папкам или закладкам, возврат к предыдущему состоянию.

В пункте *«Закладки»* создаются новые закладки и редактируются существующие. Отсюда же осуществляется переход по созданным закладкам **Konqueror**. Для этой же цели можно воспользоваться и закладками, созданными ранее в `Netscape Communicator` или `Mozilla`.

Меню *«Инструменты»* содержит следующие пункты:

- *«Выполнить команду»* — вызывает стандартный мини-терминал *KDE*,
- *«Открыть терминал»* (по умолчанию — konsole),

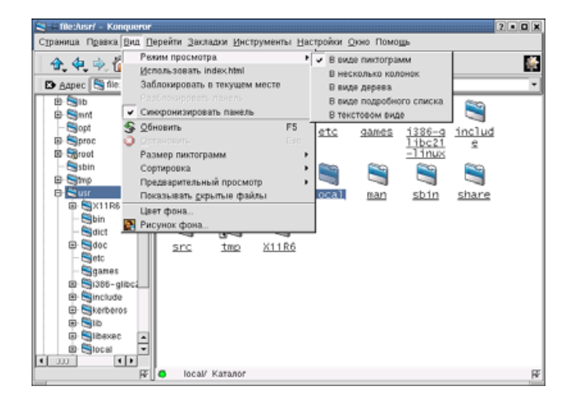


Рис. 9.5: Меню *«Вид»*

- *«Найти файл»*.

С помощью последнего вызывается панель поиска (рис. 9.7) с тремя закладками (*«Имя/Путь»*, *«Диапазон дат»*, *«Дополнительно»*), позволяющая осуществлять поиск файла по имени или маске в некоем каталоге (в том числе рекурсивно), времени создания, типу (файл, каталог, символическая ссылка и так далее), фрагменту текста.

Меню *«Настройки»*, во-первых, управляет отображением интерфейсных элементов (строки меню, инструментальной панели и т.д.). Во-вторых, оно позволяет произвести индивидуальное конфигурирование **Konqueror** и сохранить внесенные изменения, о чем будет говориться в следующем разделе.

Меню *«Окно»* ответственно отвечает за внешний вид **Konqueror**. Именно здесь можно включить или выключить дополнительные панели и определить их положение относительно текущей, показать или скрыть дерево каталогов и эмулятор терминала. Впрочем, и это предмет разговора для следующего раздела.

Наконец, меню `Помощь` включает, с одной стороны, полное руководство по **Konqueror** (к сожалению, это один из немногих элементов, не переведенных на русский язык), с другой — контекстную помощь под именем Что это. Выбрав этот пункт, а затем, зафиксировав курсор (принявший при этом знак вопроса) на какой-либо из кнопок инструментальной панели, можно не только получить описание ее назначения, но и узнать об альтернативном способе достижения того же результата, например, через главное меню (рис. 9.8).

Таковы возможности, доступные через главное меню. Возможности меню контекстного — более ограничены (рис. 9.9). Здесь можно выполнить стандартные операции вырезания, копирования и вставки, удаления и помещения

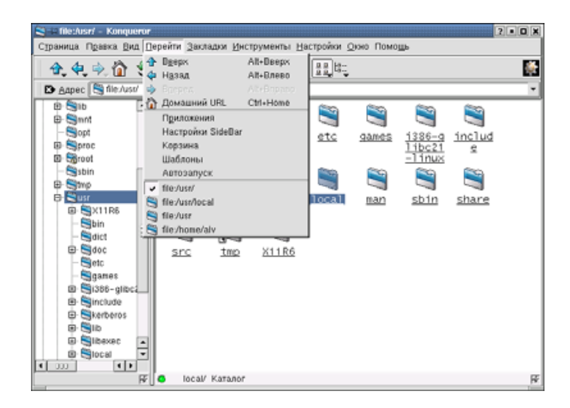


Рис. 9.6: Меню Перейти

в корзину. В зависимости от типа файла, он может быть просмотрен через предназначенную для этого встроенную (в окне **Konqueror**) или подключаемую (в самостоятельном окне) программу, либо открыт через предписанное заранее или произвольное приложение.

Кроме того, можно изменить свойства файла или каталога, начиная с его имени (рис. 9.10) и кончая правами доступа или принадлежности. Возможно и рекурсивное изменение свойств, если отметить соответствующий переключатель (рис. 9.11). Переименование файла осуществляется также только через панель свойств.

Список доступных для просмотра форматов заслуживает отдельного рассмотрения.

Разумеется, распознается текстовый формат, файлы текстового процессора Lyx раскрываются в виде исходного текста TEX. Из растровых графических форматов доступны для просмотра практически все распространенные — TIF, GIF, JPEG, PNG. Щелчок на html-файле автоматически вызывает встроенный браузер.

Привлекает внимание возможность работы с архивами и пакетами. Внутренними средствами **Konqueror** распознает архивы `*.tar.gz`, `*.tar.bz2`, `*.tgz`, позволяя обращаться с ними, как с каталогами — копировать, перемещать, удалять их содержимое целиком или частично, отдельными файлами или вложенными подкаталогами, просматривать содержимое архивированных текстовых файлов и изображений с помощью встроенной программы просмотра.

Прочие архивные форматы (`*.zip`, `*.rar` и т.д.) не распознаются внутренними средствами. Однако щелчок мышью на имени такого файлы вызывает

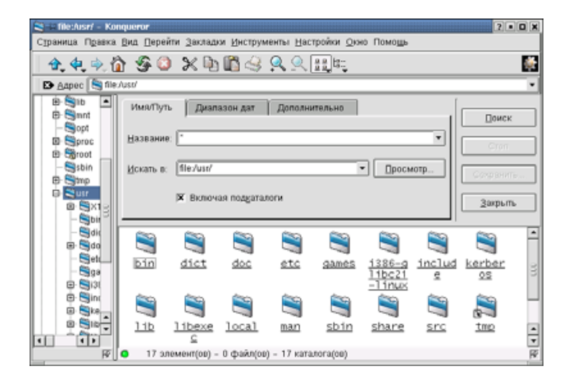


Рис. 9.7: Панель поиска файлов в konqueror и ее закладки

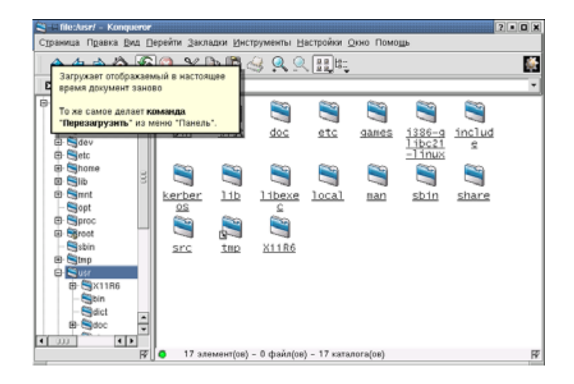


Рис. 9.8: Система контекстной помощи в **Konqueror**

программу **Ark** — штатный архиватор *KDE*, с помощью которого они и могут быть распакованы, дополнены, обновлены, как об этом говорилось выше.

Аналогично и обращение с пакетами `rpm`: щелчок на имени пакета приводит к запуску `kpackage` — средства для работы с rpm-пакетами, входящего в стандартный комплект *KDE*.

Манипулирование мышью вполне обычно для пользователей. Одинарный щелчок левой кнопкой на имени (или пиктограмме) каталога открывает его, на имени файла — вызывает просмотр файла для известных типов файла или вызывает панель выбора приложения для его открытия. Та же панель вызывается и щелчком средней клавиши мыши (рис. 9.12). Приложение можно выбрать из списка или ввести его имя непосредственно в командной строке (поддерживающей историю введенных ранее команд через выпадающее

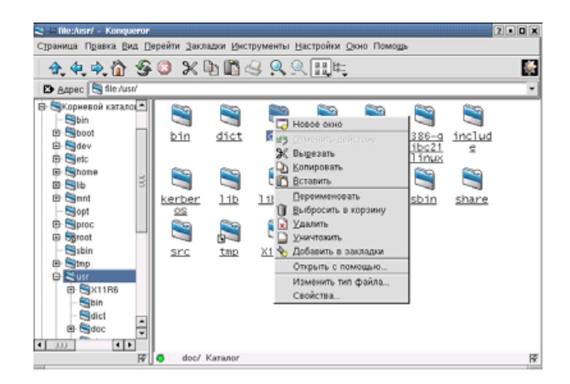


Рис. 9.9: Контекстное меню **Konqueror**

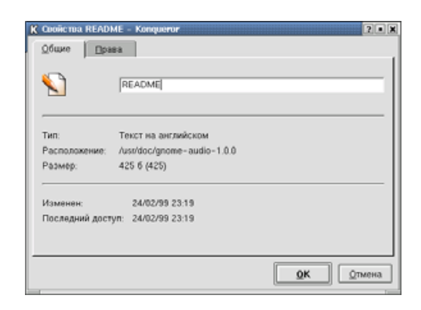


Рис. 9.10: Панель свойств файла: общие

меню).

Правая клавиша мыши вызывает контекстное меню, о котором уже говорилось.

Мышь — один из главных инструментов для копирования и перемещения файлов, что осуществляется методом `Drag'n'Drop`. Это можно делать между панелями, между панелью и деревом каталогов в любом направлении: перетаскивание файла вызывает появление контекстного меню с пунктами копирования, перемещения или создания символической ссылки.

Более того, метод `Drag'n'Drop` работает также и между панелями и эмулятором терминала, правда только в одном направлении. При перетаскивании файла из панели в область приглашения командной строки можно увидеть меню из двух пунктов — *«Вставить»* (помещающую в командную строку полный путь до перетаскиваемого файла) и Перейти (что приводит к переходу в каталог, содержащий данный файл, аналогично команде `cd`).

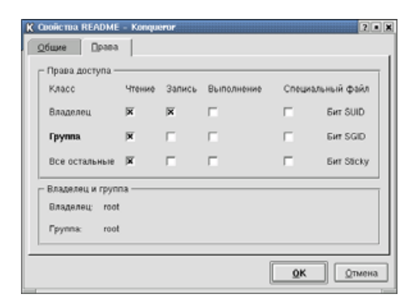


Рис. 9.11: Панель свойств файла: права доступа

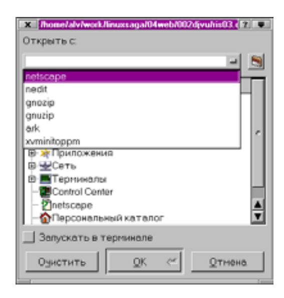


Рис. 9.12: Панель выбора приложения для открытия файла

9.3 Настройки файлового менеджера

Для настроек **Konqueror** служат три пункта главного меню: *«Вид»*, *«Настройки»*, *«Окно»*, а для их сохранения за пределами текущего сеанса требуется обратиться к двум пунктам *«Настройка»* и *«Окно»*.

В меню *«Настройка»*, помимо уже упомянутых переключателей для показа/скрытия интерфейсных элементов (меню, инструментальной панели и т.д.) можно видеть пункт собственно *«Настроить* **Konqueror**», вызывающий панель настроек с двумя фреймами: пунктами настройки слева и собственно настройками — справа. Выбор в левом фрейме пункта *«Файловый менеджер»* konqueror вызывает в правом фрейме четыре закладки.

Первая из них — *«Поведение»* (рис. 9.13) определяет условия перемещения по каталогам (открывать каталог в новом окне или в том же самом), а также указывает путь к домашнему каталогу пользователя.

Вторая закладка — *«Общий вид»* — наиболее важна (рис. 9.14). Именно здесь определяется гарнитура шрифта в панелях **Konqueror** (однако не в дереве каталогов, там шрифт можно переопределить только через системные

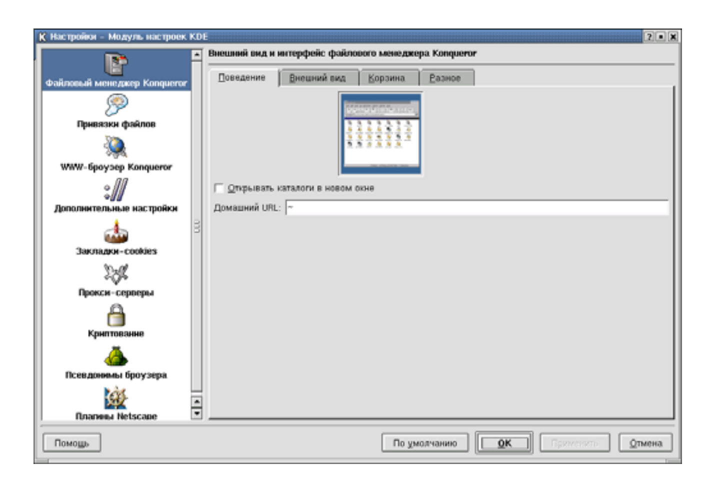


Рис. 9.13: Настройка файлового менеджера **Konqueror**, закладка *«Поведение»*

настройки *KDE*), его размер (в трех относительных градациях — *«Малый»*, *«Средний»*, *«Большой»*) и цвет. Тут же указывается, следует ли использовать переносы в подписях к пиктограммам и подчеркивать ли имена файлов в стиле web-браузеров.

Третья закладка — *«Корзина»* (рис. 9.15). Она определяет, следует ли запрашивать подтверждение на удаление, уничтожение файла и помещение его в корзину.

Наконец, закладка *«Разное»* определяет программу эмуляции терминала, вызываемого из главного меню *«Инструменты»* (рис. 9.16).

Пункт *«Привязки файлов»* в левом фрейме позволяет (рис. 9.17) закрепить за каждым из известных типов файлов одно или несколько (в порядке приоритетов, определяемых пользователем) приложений для их редактирования, а также подключать внешние программы просмотра для форматов, не воспринимаемых напрямую встроенными средствами **Konqueror**.

Дальнейшие настройки относятся к браузеру **Konqueror**, о чем будет сказано ниже. Через главное же меню Настройки можно переопределить горячие клавиши. Как уже говорилось, ими продублированы многие действия, доступные через главное или контекстное меню. Кроме того, любой из манипуляций в konqueror может быть приписана своя клавишная комбинация — из клавиш `Shift`, `Ctrl` и `Alt` в любых сочетаниях друг с другом и с прочими клавишами (рис. 9.18). Разумеется, и существующие клавишные комбинации могут быть переопределены произвольным образом.

Наконец, панели инструментов **Konqueror** настраиваются через одноименный

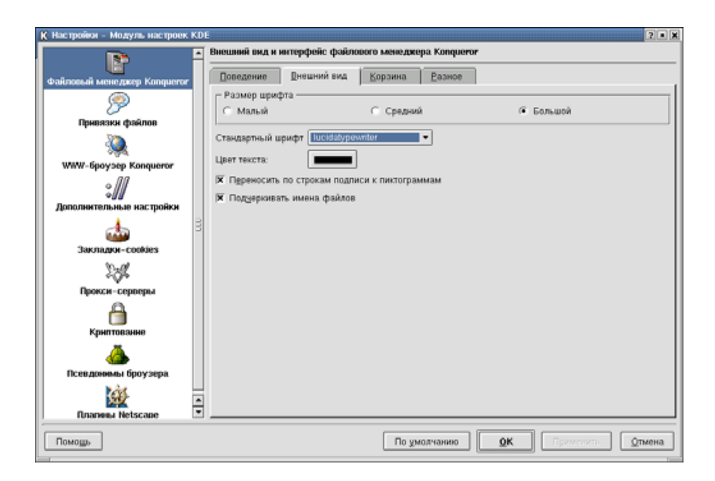


Рис. 9.14: Настройка файлового менеджера **Konqueror**, закладка *«Общий вид»*

пункт меню. Кроме главной (присутствующей по умолчанию) панели инструментов может быть включена также панель закладок. Смысл действий очевиден из рис. 9.19: в любую из этих панелей могут быть включены пиктограммы любых доступных действий и в любом порядке.

Произведя все нужные изменения, следует сохранить их для последующих сеансов — через меню *«Настройки»*- *«Записать настройки»*. При этом будет сохранено и текущее состояние переключателей из этого же меню, таких как *«Показать меню»*, *«Показать инструменты»* и т.п.

Однако сохранение настроек не окажет никакого действия на установленные через пункты меню *«Вид»* цвета фона, фоновое изображение, режим просмотра, формат списка файлов, количество и расположение панелей, показ дерева каталогов и эмулятора терминала. Для их фиксации в последующих сеансах следует обратиться к меню *«Окно»*, где выбрать пункт *«Сохранить профиль»* панели Управление файлами.

Профилей панелей по умолчанию четыре — *«Midnight Commander»* (который делает **Konqueror** похожим на `Norton Commander`, см. рис. 9.2), *«Предварительный просмотр файлов»*, *«Просмотр WWW»* (то есть профиль браузера, о котором — ниже) и *«Управление файлами»*. Однако можно создать любое их количество в соответствии с выполняемыми задачами — для этого нужно только произвести изменения в меню *«Вид»* и *«Окно»* и приписать профилю некоторое имя (рис. 9.20).

Через сохранение профиля фиксируется и порядок сортировки, который можно определить (если режим просмотра текущей панели текстовый или списоч-

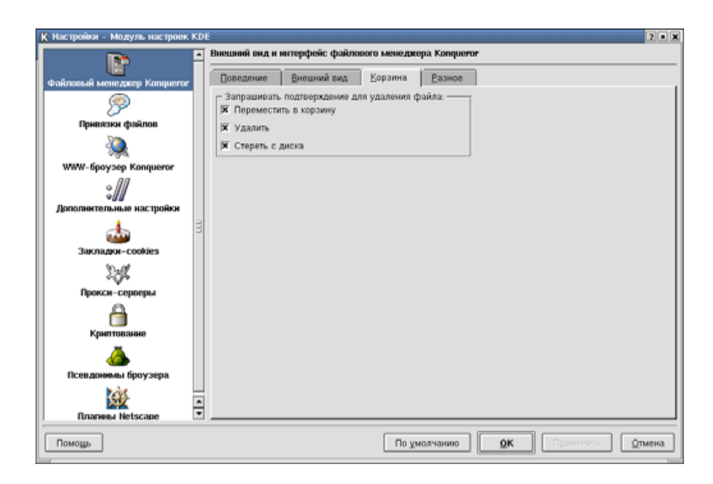


Рис. 9.15: Настройка файлового менеджера **Konqueror**, закладка *«Корзина»*

ный), щелкнув на заголовке требуемой колонки (например, на имени файла, размере и т.д.).

9.4 Браузер — настройки и возможности

Все сказанное выше относилось к файловому менеджеру **Konqueror**. Однако он же выступает и в роли браузера. Переход в этот режим браузера осуществляется в трех случаях: при прямом щелчке левой кнопки мыши на html-файле, при переходе в каталог, содержащий файл `index.html`, если включено использование индексного файла (через меню *«Вид»*), и при загрузке профиля панелей Просмотр WWW (рис. 9.21).

Браузер **Konqueror** отличает чрезвычайное быстродействие. Возможности же его практически такие же, как и у любого современного браузера. Он корректно воспроизводит html-документы с символами кириллицы, различая кодировки русского языка: koi8-r, cp1251, iso8859-5. Переключаются кодировки через меню *«Вид»*, где в режиме браузера появляется пункт *«Выставить кодировку»*.

Впрочем, если остановиться на автоматическом ее определении, русские тексты также будут в большинстве случаев читаться нормально, вне зависимости от того, есть ли в html-файле прямое указание на кодировку страницы или нет.

Кроме того, **Konqueror** распознает традиционные для Интернет графические

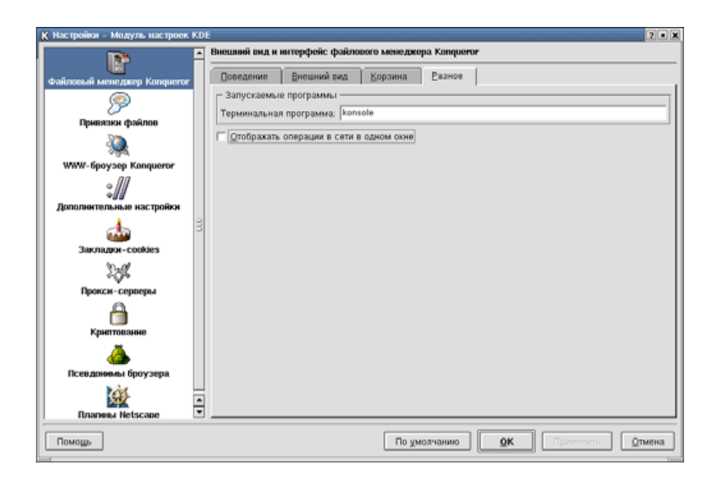


Рис. 9.16: Настройка файлового менеджера **Konqueror**, закладка *«Разное»*

форматы — GIF, JPEG, PNG. Правда, пока нет дополнительного модуля для просмотра DjVu, однако это можно сделать через внешнюю программу. В таком качестве можно использовать `Netscape Navigator` или `Mozilla` с соответствующим plugin.

К внешней программе придется прибегнуть и для воспроизведения RealAudio, RealVideo, MPEG-файлов и файлов Macromedia Flash и Shockwave.

Аплеты Java, также как и JavaScript, **Konqueror** поддерживает в полном объеме, хотя по умолчанию обе эти возможности отключены. Поддерживаются также каскадные таблицы стилей (CSS) и плавающие фреймы (*«iframe»*). Кроме того, **Konqueror** имеет полноэкранный режим просмотра web-страниц.

Браузер **Konqueror** дает очень простой доступ к редактированию html-кода: для этого достаточно выбрать в меню Вид пункт Просмотреть источник. Файл html будет открыт в том редакторе, который внесен в список приоритетных приложений для документов HTML при определении ассоциации файлов (см. рис. 9.16). Кроме того, просмотреть или отредактировать код html можно и через контекстное меню, выбрав в нем пункт *«Открыть с»* и затем одно из приоритетных приложений.

Настройки браузера производятся через меню *«Настройки»- «Настроить konqueror»*. В левом фрейме настроечной панели выбирается пункт *«WWW-браузер Konqueror»*. В этой панели пять закладок: *«HTML»*, *«Внешний вид»*, *«Java»*, *«JavaScript»*, *«Plugins»*. В первой, помимо подчеркивания и изменения курсора, можно отключить автоматическую загрузку графики (рис. 9.22).

Закладка *«Внешний вид»* позволяет выбрать подходящие шрифты для разных

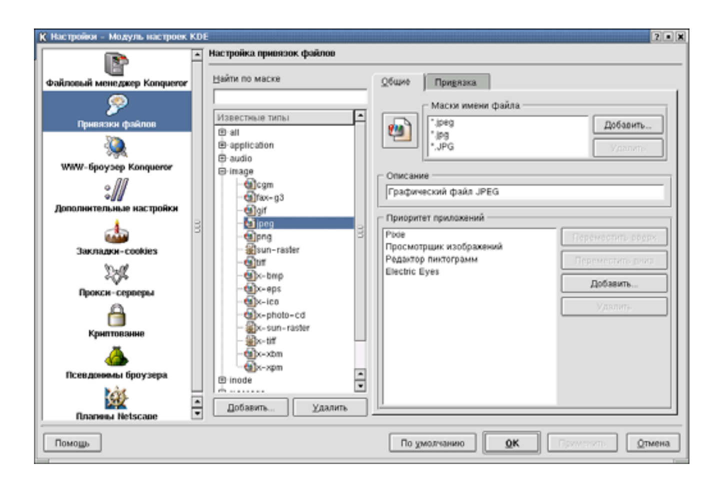


Рис. 9.17: Настройка привязки файлов

текстовых элементов, задать его абсолютный минимальный размер и размер относительный (*«Маленький»*, *«Средний»*, *«Большой»*), а также указать кодировку по умолчанию и шрифтовые гарнитуры для различных текстовых элементов (рис. 9.23).

Закладки *«Java»* и *«JavaScript»* позволяют не только включить поддержку соответствующих языков глобально, но дать индивидуальные установки для конкретных доменов. В закладке же *«Plugins»* можно включить или выключить поддержку дополнительных модулей.

Браузер **Konqueror** может использоваться как своего рода мета-поисковая

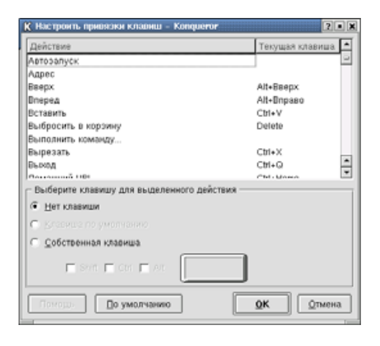


Рис. 9.18: Определение комбинаций горячих клавиш

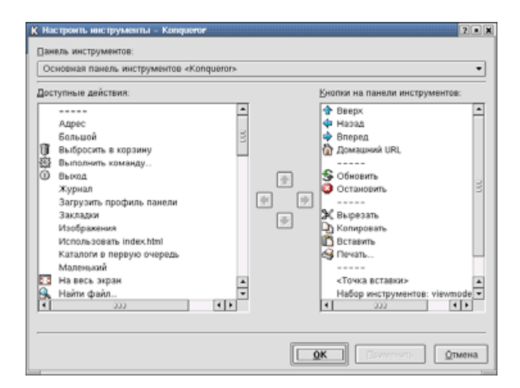


Рис. 9.19: Настройка инструментальных панелей

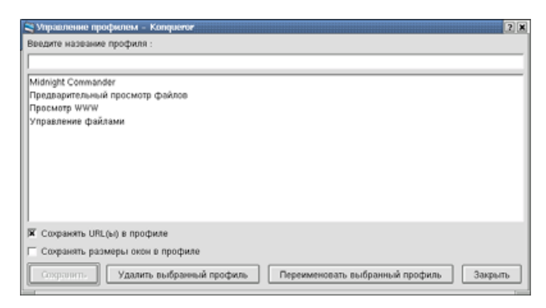


Рис. 9.20: Управление профилями панелей

машина. Для этого в пункте *«Настройки»- «Ключевые слова Интернет»* приведен обширный список доступных поисковых машин, который может быть отредактирован и расширен. Наконец, могут быть настроены также правила обработки `cookies` и, при необходимости, параметры прокси-сервера.

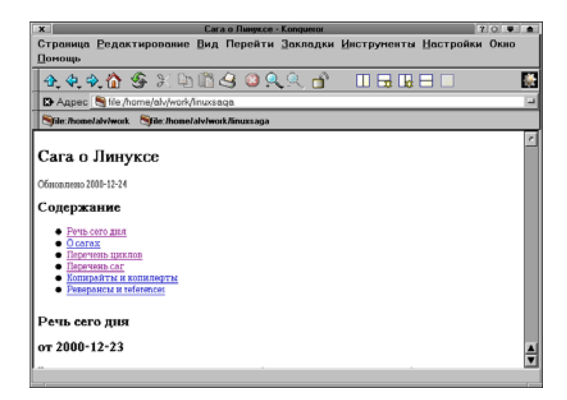


Рис. 9.21: **Konqueror** в качестве браузера

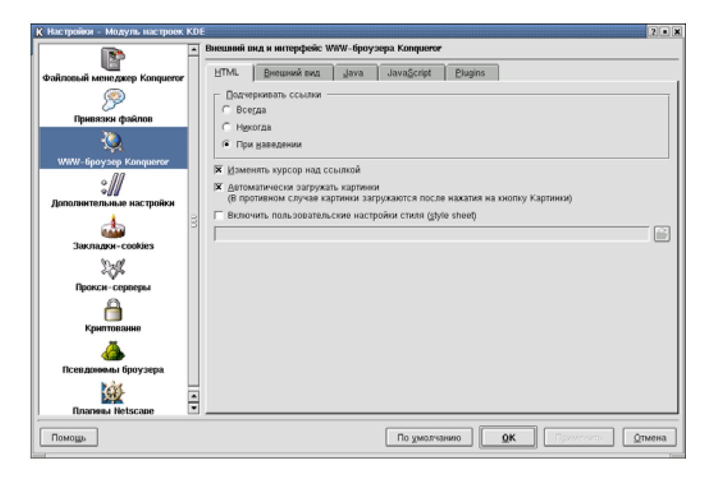


Рис. 9.22: Настройка браузера **Konqueror**, закладка *«HTML»*

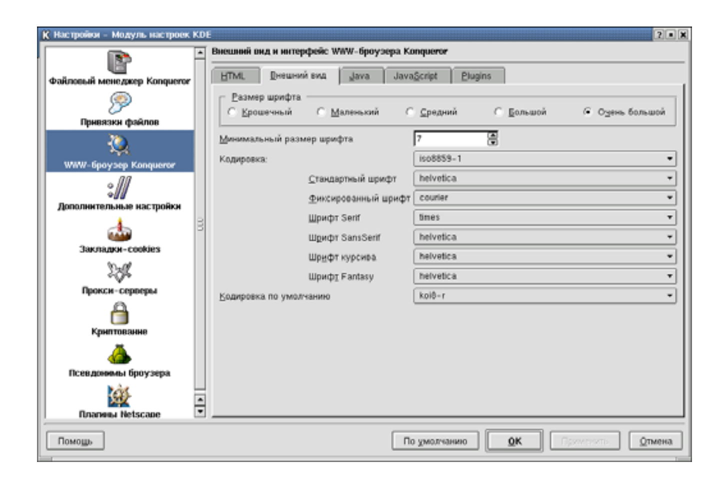


Рис. 9.23: Настройка браузера **Konqueror**, закладка *«Внешний вид»*

Глава 10

Текстовые редакторы

Текстовые редакторы занимают особое место среди приложений Linux, являясь универсальным инструментом для решения самых разных задач. Они разделяются на две группы — редакторы командного стиля и меню-ориентированные. Первые, работающие в консольном режиме, — в сущности, интерпретаторы собственного командного языка, обнаруживающие родство с командными оболочками типа bash.

Все действия в них выполняются не путем манипулирования пунктами меню или мышью, а подачей прямых управляющих директив, за которыми, как правило, закреплены определенные клавишные комбинации. Типичными их представителями являются описанные ниже vi и joe.

Меню-ориентированные редакторы представлены как консольными программами, так и приложениями графического режима. К первым относится, например, mcedit — редактор, встроенный в файловый менеджер Midnight Commander, но могущий использоваться и самостоятельно. Примером развитого текстового редактора для системы X Window System является kwrite, входящий в состав интегрированной среды *KDE*.

В заключение главы будет затронута тема, тесно связанная с обработкой текстов — о кодировках кириллических текстов и их конверторах.

10.1 vi — универсальный редактор для UNIX-систем

Консольный редактор vi (или какой-либо из его клонов) — непременный атрибут любой UNIX-системы, вызываемый по умолчанию при общесистемных настройках. И потому любой пользователь Linux должен иметь о нем представление, хотя для повседневной работы какой-либо иной редактор может оказаться более подходящим.

Существует несколько редакторов, основанных на vi, включающих дополни-

тельные возможности, но полностью совместимых с ним по системе базовых команд. И потому знание vi обеспечит возможность работы с любым из его клонов. Более того, если в некоей UNIX-системе вместо vi используется какой-либо из редакторов-клонов, например, vim или elvis, в каталоге /bin всегда будет представлен файл vi, являющийся символической ссылкой на реальный исполнимый файл этого редактора. Поэтому vi или его аналог всегда может быть запущен командой

```
vi имя_файла
```

хотя реально при этом запускается, например, vim. Именно так происходит в дистрибутиве **ASPLinux** по умолчанию, где командами vi и vim запускается один и тот же редактор — vim (Vi IMproved), о котором и пойдет речь в этом разделе. Подчеркнем, что ниже термины vi и vim используются как синонимы, но везде имеется в виду vim. Хотя практически все описанные приемы относятся и к классическому vi.

Редактор vim может быть запущен из командной строки оболочки в консоли или окне эмулятора терминала X Window System, с именем файла (и указанием, при необходимости, полного пути до него) или без такового. В первом случае открывается существующий файл, если он существует, или создается новый — в текущем каталоге, или в каталоге, определенном в пути к нему. Например, команда

```
vi .bashrc
```

вызовет для редактирования конфигурационный файл оболочки bash из домашнего каталога пользователя, а команда

```
vi ~/mytext/newtext.txt
```

создаст новый текстовый файл newtext.txt в подкаталоге ~/mytetxt домашнего каталога пользователя (напомним только, что каталог ~/mytetxt должен уже существовать, иначе при записи файла последует сообщение об ошибке).

Команда vi (или vim) без имени файла откроет редактор **vim** и выведет заставку (рис. 10.1), после чего можно или начать работу (как — будет сказано ниже) или получить справку, набрав с клавиатуры

```
:help
```

или нажав клавишу F1 (последнее может не сработать в терминальном окне).

Рис. 10.1: Заставка редактора vim

Настоятельно рекомендуем воспользоваться этой возможностью (рис. 10.2), так как работа в `vi` (и в `vim`) может оказаться непривычной для пользователя, в частности, попытка немедленно начать ввод текста будет неудачной. Потому следует сначала ознакомиться с режимами работы `vi`.

В `vi` существует три принципиально различных режима работы — командный режим (`command mode`), режим ввода (`edit mode`) и т.н. ex-режим, или режим построчного редактирования (`ex mode`). Без четкого понимания этого работа в `vi` просто невозможна.

Командный режим включается по умолчанию при загрузке `vi`. Это, собственно говоря, интерпретатор встроенного языка редактора. В этом режиме нажатия клавиш приводят не к вводу символов, а интерпретируются как внутренние команды навигации и редактирования. Например, нажатие клавиши `h` вызывает перемещение курсора на один символ влево, клавиши `l` — на один символ вправо, `k` — на строку вверх, `j` — на строку вниз и т.д.

Буквенные команды командного режима работают не только при латинской, но и при кириллической раскладке клавиатуры. Однако в этом случае они должны вводиться при нажатой клавише `Alt`

Соответственно, создание текста в командном режиме невозможно. Для этого следует перейти в режим ввода, для чего служат клавиши командного режима `a` (от `append`) и `i` (от `insert`). Здесь нажатия клавиш приводят к вводу обычных буквенно-цифровых символов (после текущей позиции курсора и перед ней, соответственно), позволяя создавать новый текст или редактировать имеющийся. Хотя последнее более эффективно в командном режиме, возврат в который осуществляется клавишей `Esc`.

Для операций с документами (то есть файлами) предназначен ex-режим, вы-

```
help.txt*      For Vim version 6.0z.  Last change: 2001 Mar 17

                        VIM - main help file
                                                                     k
      Move around:  Use the cursor keys, or "h" to go left,        h   l
                    "j" to go down, "k" to go up, "l" to go right.   j
Close this window:  Use ":q<Enter>".
   Get out of Vim:  Use ":qa!<Enter>" (careful, all changes are lost!).
Jump to a subject:  Position the cursor on a tag between |bars| and hit CTRL-].
   With the mouse:  ":set mouse=a" to enable the mouse (in xterm or GUI).
                    Double-click the left mouse button on a tag between |bars|.
        jump back:  Type CTRL-T or CTRL-O (can be repeated).
Get specific help:  It is possible to go directly to whatever you want help
                    on, by giving an argument to the ":help" command |:help|.
                    It is possible to further specify the context:
                          WHAT                  PREPEND    EXAMPLE
                      Normal mode commands     (nothing)   :help x
                      Visual mode commands        v_       :help v_u
                      Insert mode commands        i_       :help i_<Esc>
                      Command-line commands       :        :help :quit
                      Command-line editing        c_       :help c_<Del>
                      Vim command arguments       -        :help -r
                      options                     '        :help 'textwidth'
"help.txt" [readonly] 178L, 6558C
```

Рис. 10.2: Встроенная система помощи редактора vim

зываемый клавишей `:` (командного режима; в режиме ввода нажатие этой клавиши вызывает ввод двоеточия). После этого дается команда ex-режима для следующих действий:

- открытия существующего файла (`:e имя_файла` — здесь и далее символ : означает команду ex-режима) если какой-либо файл перед этим уже загружен, он будет закрыт и замещен новым;
- вставки существующего файла в позицию курсора (`:r имя_файла`);
- записи файла (`:w`), в том числе под другим именем (`:w имя_файла`);
- выхода из сохраненного файла (`:q`);
- выхода из редактора `vi` с предварительным сохранением измененного файла (`:x`).

Возможно совмещение команд ex-режима, например, `:wq` — выход с предварительным сохранением измененного файла (что аналогично команде `:x`).

Команды отправляются на исполнение нажатием клавиши `Enter`, после чего происходит возврат в командный режим. Однако попытка, например, закрыть редактор без сохранения изменений в редактируемом документе (командой `:q`) или загрузить новый файл (командой `:e`), не сохранив предыдущий, вызовет сообщение об ошибке. Для принудительного выполнения таких действий команды `:q` и `:e` должны использоваться в сочетании с символом `!`. Например, команда `:q!` закроет редактор `vi`, не сохранив изменений в текущем файле.

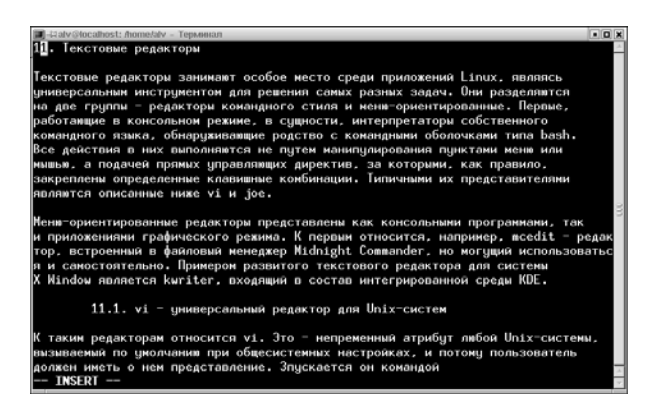


Рис. 10.3: Редактор vim в режиме ввода (обратите внимание на надпись – `INSERT` – внизу экрана)

Действия ex-режима частично дублируются в режиме командном. Так, для закрытия же файла (с предварительным сохранением изменений) используется последовательность `Z` `Z` командного режима, что является эквивалентом команды `:wq` ex-режима.

Определение текущего режима в `vim` несколько усовершенствовано по сравнению с исходным `vi`. Переход в режим ввода индицируется появлением надписи `INSERT` внизу экрана (рис. 10.3). При возврате в командный режим (нажатием клавиши `Esc`) она исчезает. Повторное нажатие `Esc` в командном режиме вызывает звуковой сигнал, служащий косвенной его индикацией. Нахождение в ex-режиме фиксируется символом `:` в нижней части экрана.

Такая система работы может показаться запутанной начинающему пользователю. Однако она имеет глубокое внутреннее обоснование. Редактор `vi` создавался изначально как кросс-платформенный, который обязан работать на любых типах реальных и виртуальных терминалов. И потому все действия в нем можно осуществить, не покидая основной, алфавитно-цифровой, части клавиатуры, без обращения к дополнительным клавишам — стрелкам управления курсором, `Home`, `End`, `PageUp`, `PageDown`, `Ins`, `Del`, `Backspace`.

Это, с одной стороны, обеспечивает быстроту и эффективность работы (правда, только при наличии доведенных до автоматизма навыков). С другой стороны, такая особенность `vi` позволяет не заботиться о настройках виртуальных терминалов в X Window System, которые существенно влияют на поведение дополнительных клавиш: внутренние команды навигации и редактирования всегда будут интерпретироваться идентично.

Поскольку командный режим `vi` является основным для этого редактора, имеет смысл ознакомиться с ним подробнее. Все изобилие команд `vi` (а их

насчитывается много десятков) можно разделить на три группы:

- команды навигации,
- команды редактирования,
- команды перехода (в режим ввода).

Команды навигации служат для перемещения курсора по тексту. В консольном режиме **ASPLinux** для этого могут быть использованы и обычные клавиши управления курсором (стрелки, PageUp/PageDown, Home/End). Однако уже в различных программах эмуляции терминала в графическом режиме их поведение неоднозначно и зависит от настроек, а в других UNIX-системах может быть иным. К тому же внутренние команды дают больше возможностей для навигации по тексту, чем клавиши стандартных клавиатур.

Так, в `vi` можно использовать клавиши h и l, k и j, действие которых эквивалентно нажатиям клавиш Left и Right, Up и Down, соответственно. Но, кроме того, с помощью парных клавиш w и W можно переместиться вперед, соответственно, на т.н. «маленькое» слово (то есть отделенное пробелом или любым знаком препинания, символами - или +) и на «большое» слово (то есть обязательно отделенное пробелом). Пара клавиш b и B выполняет аналогичное перемещение назад, а клавиши e и E перемещают курсор в конец следующего «маленького» или «большого» слова.

Вообще, для многих команд `vi` характерно наличие парных эквивалентов — в нижнем и верхнем регистрах одной клавиши (w и W, e и E); действие второй команды из пары как бы расширяет действие первой.

Возможны также перемещения в предыдущее (() и последующее ()) предложение, в начало (H) и конец (L) экрана, в начало (0 — ноль) и конец ($) строки и т.д. — список навигационных команд приближается к 30. Иными словами, нажатием одной клавиши или, в крайнем случае, двухклавишной комбинации (Ctrl+F - на следующую экранную страницы, Ctrl+B — на предыдущую) можно переместиться в абсолютно любое заранее определенное место текстового документа.

Команды навигации `vi` могут использоваться с численными аргументами.

Например, команда `5h` переместит курсор на 5 символов влево (считая символ в позиции курсора, а команда `3k` — на три строки вверх.

Для навигации по тексту могут использоваться клавиши - (минус) для перемещения на одну строку вверх и + (плюс) для перемещения на одну строку вниз. Особенно эффективны они в сочетании с численными аргументами: последовательность 7 + перемещает курсор на семь строк вперед, а 1 3 - — на тринадцать строк назад (в обоих случаях включая строку в позиции курсора).

Команды редактирования предназначены для модификации существующего текста без перехода в режим ввода. Конечно, и в последнем возможно удаление и замена символов стандартными клавишами `Del` или `Backspace`, но, как и в случае с навигацией, возможности командного режима в этом отношении много шире.

Так, наряду с удалением единичного символа в позиции курсора (`x`) или перед ней (`X`), возможно удаление слова (`d` `w`), строки (`d` `d`) или ее части перед (`d` `D`) или после (`d` `G`) курсора, предложения (`d` `)`) или абзаца (`d` `}`).

Как и навигационные команды, команды редактирования могут использоваться с численными аргументами. Так, последовательность `5` `d` `d` удалит текущую строку и еще пять строк вниз. А с помощью последовательности `3` `d` `w` можно удалить три слова подряд (включая то, на котором находится курсор).

Не меньше команд отвечает за копирование фрагментов, их вставку и замену. Например, последовательность `y` `w` копирует «маленькое» слово, `y` `W` —«большое» `y` `y` - строку, `y` `)` — предложение, `y` `}` — абзац. Клавишей же `p` удаленный или скопированный фрагмент вставляется в позицию курсора.

Действие ошибочно введенных команд редактирования может быть отменено клавишей `u` (undo). Вторичный ввод этой команды приведет к отмене предыдущего действия, и так далее. Для возврата ошибочно отмененной операции используется комбинация клавиш `Ctrl`+`R`.

Описание всех команд редактирования заняло бы слишком много места. Подчеркнем лишь, что, как и в случае с навигацией, нажатием одной-двух клавиш можно удалить, скопировать, вставить и переместить текстовый блок практически любого размера — от единичного символа до произвольного их количества (строки, предложения, абзаца, экранной страницы). Команды перехода могут рассматриваться как подмножество команд редактирования, после которых возможен ввод новых символов и их последовательностей. Кроме уже упомянутых `a` и `i` (ввод после и перед курсором), к ним относятся:

- `A` — ввод текста в конец строки,
- `I` — ввод в начало строки,
- `o` — создание новой строки под текущей с возможностью ввода в нее текста,
- `O` — создание новой строки над текущей, в которую также можно ввести текст.

Как и большинство прочих команд редактора vi, команды перехода могут использоваться с числовыми аргументами. Так, если нажать последовательно `3` `a` и после этого ввести некоторую последовательность символов, по

выходе в командный режим (клавишей Esc) она будет повторена трижды. Аналогично, ввод текста после набора числа (клавиши 0-9) с последующим нажатием O введет указанное количество идентичных строк.

Редактор vi располагает средствами поиска и замены текстовых фрагментов, в том числе и с использованием регулярных выражений. Для этого предназначена команда ex-режима :s (substitute). Она дается в форме

```
:999s/text1/text2/опция
```

где 999 — количество строк, в которых операции поиска и замены должны осуществляться (без указания его действие команды распространяется только на текущую строку). Если поиск и замену необходимо выполнить по всему тексту документа, дается команда :%s.

Следует подчеркнуть, что если не указать заменяющего текста (и опций замены), все текстовые фрагменты, указанные в качестве заменяемого текста, будут просто удалены. Во избежание этого используются опции команды :s. Так, опция /c предписывает запрос подтверждения на замену для каждого найденного фрагмента.

И поиск, и замена в vi возможны только для последовательности символов, составляющих одну строку (то есть не содержащей символа LF). Заменяющая последовательность символов также должна образовывать единую строку.

Возможности редактора vi описанным не исчерпываются. В частности, он (вернее, его клон vim) поддерживает язык макрокоманд и допускает их протоколирование. В комплекте с редактором (в каталоге /usr/share/vim/vim60z) имеется большое количество таких макросов. Там же находится и детальная документация по их применению.

10.2 Текстовый редактор joe

Текстовый редактор joe — типичный представитель консольных редакторов командного стиля. В отличие от vi, он имеет более простую систему команд и предоставляет ряд дополнительных возможностей, в частности, удобные и развитые средства создания пользовательских макросов. Кроме того, joe способен эмулировать команды других распространенных на UNIX-платформах текстовых редакторов — vi, emacs, pico и кроссплатформенного редактора **WordStar**.

Редактор joe представлен в пяти вариантах: собственно joe, rjoe — версия с ограниченными возможностями, jmacs — эмулятор редактора emacs, jpico — эмулятор редактора pico, jstar — эмулятор редактора **WordStar**, старейшего из полноэкранных текстовых редакторов, система команд кото-

```
   IWA  work/articles/other/joe/joe  Row 1    Col 1    10:57  Ctrl-K H for help
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<html>
<body>
<title></title>

<h1>Текстовый редактор Joe
<br>как типичный представитель команяных редакторов</h1>

<p>Тестовые процессоры занимают особое место в мире Unix- и Unix-подобных
систем. отдаленно сравнимое только с ролью текстовых процессоров в мире
DOS/Windows. И потому представлены они многими множествами видов и
разновидностей. Среди которых почетное место занимает текстовый редактор
Joe. который и будет предметом рассмотерния в предлагаемой серии заметок.

<h2>Вводные замечания</h2>

<p>Текстовый редактор Joe - типичный представитель консольных редакторов
команданого стиля. то есть ориентированных не на действие через меню. а на
управление с помощью прямых директив. В чем причина моего обращения к этой
теме? Попробую объяснить.

<p>О назначении консольных текстовых редакторов и их необходимости даже в
** Joe's Own Editor v2.81 ** Copyright (C) 1995 Joseph H. Allen **
```

Рис. 10.4: Вид редактора joe по умолчанию

рого в дальнейшем использовалась во многих интегрированных средах программирования.

Каждая из этих версий, кроме `rjoe`, представляет полнофункциональный текстовый редактор со своими особенностями. Но ниже будет описан только собственно `joe`.

Запускается `joe` из командной строки оболочки в консоли или эмуляторе терминала одноименной командой, можно с именем файла, предназначенного для редактирования. В случае если файла с таким именем не существует, создается новый пустой файл.

Кроме этого, при запуске `joe` можно использовать ряд опций командной строки. Представление о них дает чтение страниц экранной документации (`man joe`).

После запуска `joe` выглядит следующим образом (рис. 10.4): экран с текстом, строка заголовка в верхней его части и строка с указанием авторства — в нижней.

Подсказка строки заголовка указывает на способ выведения системы помощи (комбинацией клавиш `Ctrl`+`K`+`H`) и просмотра ее: посредством `Esc`+`.` — вперед или `Esc`+`,` — назад (рис. 10.5).

Изучение страниц помощи дает представление о возможностях редактора для навигации по тексту и его редактированию. Основными средствами для этого являются внутренние команды `joe`, выполняемые в большинстве случаев нажатием комбинации клавиш `Ctrl` (изредка - `Esc`)+`СИМВОЛЬНАЯ КЛАВИША`, что эквивалентно переходу в командный режим `vi`. Навигация и редактирование могут выполняться и стандартными клавишами PC-клавиатуры, однако использование внутренних команд, как и в случае с `vi`, предпочтительнее.

```
rxvt
   Help Screen    turn off with ^KH    more help with ESC . (^[.)
CURSOR               GO TO                BLOCK          DELETE     MISC          EXIT
^B left ^F right     ^U  prev. screen     ^KB begin      ^D char.   ^KJ reformat  ^KX save
^P up    ^N down     ^V  next screen      ^KK end        ^Y line    ^T  options   ^C  abort
^Z previous word     ^A  beg. of line     ^KM move       ^W >word   ^R  refresh   ^KZ shell
^X next word         ^E  end of line      ^KC copy       ^O word<   ^@  insert    FILE
SEARCH               ^KU top of file      ^KW file       ^J >line   SPELL         ^KE edit
^KF find text        ^KV end of file      ^KY delete     ^_ undo    ^[N word      ^KR insert
^L  find next        ^KL to line No.      ^K/ filter     ^^ redo    ^[L file      ^KD save
    IWA  work/articles/other/joe/joe  Row 1    Col 1    11:00  Ctrl-K H for help
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<html>
<body>
<title></title>

<h1>Текстовый редактор Joe
<br>как типичный представитель команднык редакторов</h1>

<p>Тестовые процессоры занимают особое место в мире Unix- и Unix-подобных
систем, отдаленно сравнимое только с ролью текстовых процессоров в мире
DOS/Windows. И потому представлены они многими множествами видов и
разновидностей. Среди которых почетное место занимает текстовый редактор
Joe, который и будет предметом рассмотерния в предлагаемой серии заметок.
```

Рис. 10.5: Редактор joe: первая страница системы помощи

Кроме преимущества в скорости, это обеспечивает идентичность действий на любых типах терминалов.

Мышь в `joe` поддерживается стандартным для Linux-консоли образом, то есть не как указательное устройство, а как средство выделения и копирования экранных блоков. Это относится и к текстовой консоли (даже при подключении `gpm`), и к эмуляторам терминалов графического режима.

Редактор `joe` допускает выделение блоков и отдельных знаков, их копирование, перемещение и удаление, форматирование абзацев (центрирование, лево- и правостороннее выравнивание и т.д.), вставку существующих файлов в текущий документ и запись выделенных фрагментов в виде отдельных файлов (см. рис. 10.5).

Редактор `joe` имеет функцию многоуровневой отмены и возврата отмененных операций, позволяет подключить внешнюю программу проверки орфографии, такую как `aspell`, в том числе и для русскоязычных текстов. Имеются достаточно развитые средства поиска и замены, в том числе с использованием шаблонов и регулярных выражений. Есть возможность создания закладок (`Bookmarks`) и перехода к ним.

В `joe` имитируется многооконный режим: поле текущего документа может быть разбито пополам, и далее каждое из них также может делиться сколь угодно дробно (правда, только по горизонтали). Обеспечена также одновременная работа со многими документами, каждый из которых может быть выведен в оконном или полноэкранном виде (рис. 10.6).

Количество одновременно открытых файлов, ограничено только ресурсами компьютера. Более того, `joe` позволяет работать с документами, объем которых превышает объем `всей` доступной (то есть физической + виртуальной)

```
   Help Screen    turn off with ^KH    prev. screen ^[,    next screen ^[.
 ^KO Split the window in half          ^KE Load file into window
 ^KG Make current window bigger        ^KT Make current window smaller
 ^KN Go to the window below            ^KP Go to the window above
 ^C  Eliminate the current window      ^KI Show all windows / Show one window
    IWA  work/articles/other/joe/joe  Row 1    Col 1    7:48  Ctrl-K H for help
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<html>
<body>
<title></title>

<h1>Текстовый редактор Joe
<br>как типичный представитель командных редакторов</h1>

    I    /home/alv/.joerc             Row 1    Col 1    7:48  Ctrl-K H for help

                        Initialization file for JOE
                               Standard Joe

 JOE looks for this file in:
        1 - .joerc
        2 - $HOME/.joerc
        3 - /usr/local/lib/joerc
```

Рис. 10.6: Работа с двумя документами в оконном режиме

памяти.

В joe поддерживается собственный макроязык с достаточно прозрачным синтаксисом. Кроме того, имеется режим протоколирования макросов, что позволяет быстро наращивать его возможности.

Редактор joe обладает системой интерактивной настройки ряда параметров, таких как перенос слов, абзацный отступ и т.д. (рис. 10.7). Правда, установки эти действуют только в текущем сеансе. Для перманентных изменений необходимо редактирование конфигурационного файла, о чем будет сказано ниже.

Редактор joe корректно работает с кириллицей при правильно русифицированной консоли. Более того, все клавишные комбинации работают и при латинской, и при русской раскладке клавиатуры. Правда, в последнем случае часто требуется дополнительное нажатие управляющей клавиши Ctrl.

Для эффективного использования joe следует четко уяснить, что это командный редактор в чистом виде. То есть все действия по редактированию текста осуществляются соответствующими встроенными командами, к которым привязаны комбинации клавиш. В сущности, как будет показано ниже, все это — макросы на собственном языке joe. Из чего следует, что, с одной стороны, система команд может быть сколь угодно наращена, с другой — что клавишные комбинации для них могут быть переопределены произвольным образом.

Структура закрепленных за командами по умолчанию клавишных комбинаций проста и очень логична. За самыми простыми и частыми действиями для навигации и редактирования закреплены двухклавишные комбинации. Это, как правило, сочетание одновременно нажатых клавиш Ctrl и буквенной, имеющей обычно мнемонический смысл.

Полный список встроенных команд и привязанных к ним клавишных комбина-

```
  Help Screen    turn off with ^KH    prev. screen ^[.    next screen ^[.
MACROS          MISC               SCROLL     SHELL         GOTO         I-SEARCH
^K[ 0-9 Record  ^K SPACE status    ^[W Up     ^K' Window    ^[B To ^KB   ^[R Backwards
^K]     Stop    ^K\ Repeat         ^[Z Down   ^[! Command   ^[K To ^KK   ^[S Forwards
^K 0-9  Play    ^[M Math           ^K< Left   QUOTE         DELETE       BOOKMARKS
^K?     Query   ^KA Center line    ^K> Right  `  Ctrl-      ^[Y yank     ^[ 0-9 Goto
^[D     Dump    ^[H Message                   ^\ Meta-      ^[O line<    ^[^[   Set
   IWA  work/articles/other/joe/joe   Row 1    Col 1    8:21  Ctrl-K H for help
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<html>
<body>
<title></title>

<h1>Текстовый редактор Joe
<br>как типичный представитель командных редакторов</h1>

<p>Тестовые процессоры занимают особое место в мире Unix- и Unix-подобных
систем, отдаленно сравнимое только с ролью текстовых процессоров в мире
DOS/Windows. И потому представлены они многими множествами видов и
разновидностей. Среди которых почетное место занимает текстовый редактор
Joe, который и будет предметом рассмотерния в предлагаемой серии заметок.

<h2>Вводные замечания</h2>
T Overtype OFF          I Autoindent ON          Word wrap ON
```

Рис. 10.7: Окно помощи с дополнительными возможностями (вверху), система интерактивной настройки (внизу)

ций можно посмотреть в экранной документации. Приведем только основные примеры.

Так, комбинация Ctrl+B (от backward) перемещает курсор на один знак влево, Ctrl+F (от forward) — на один знак вправо, Ctrl+Z — переход к предыдущему слову, Ctrl+X — к последующему слову и так далее (см. рис. 10.5).

В некоторых случаях в качестве управляющей клавиши используется Esc: если после ее нажатия набрать литеру W — курсор переместится на строку вверх, литеру Z — на строку вниз. Кроме того, Esc служит для вызова проверки правописания для слова (Esc + N) и всего файла (Esc + L). Нажатие Esc два раза подряд приводит к установке закладки (bookmark), которая маркируется произвольным числом, а Esc плюс этот номер вызывает переход к установленной закладке. Правда, очевидно, что закладок не может быть больше 10; и к тому же по завершении сеанса они не сохраняются.

Все клавишные комбинации в joe не чувствительны к регистру, причем не только для буквенных, но и символьных клавиш. Так, для отмены последней операции (как уже говорилось, многоуровневой) зарезервирована комбинация Ctrl+_ (знак подчеркивания), а для возврата отмененного действия — Ctrl+^; однако можно использовать также их аналоги для нижнего регистра — (дефис или минус) в первом случае, и 6 — во втором.

Кроме того, двухклавишные комбинации не чувствительны и к раскладке клавиатуры: сочетание Ctrl+T (латинское) будет вызывать систему настройки joe и при кириллической раскладке. Интересно, что для пролистывания страниц помощи (показанных, например, на рис. 10.2, рис. 10.3 и рис. 10.3) вперед и назад при кириллической раскладке следует нажимать Esc и .

(или, соответственно, запятую) также в ее положении на русифицированной клавиатуре (то есть в нижнем правом углу для win-клавиатур и на верхнем регистре цифр 5 и 7 — для DOS-раскладок).

Для более сложных или редких действий используются трехклавишные комбинации.

Это почти исключительно одновременно нажатые Ctrl+K, после чего нажимается литерная клавиша. Так, операции с блоками осуществляются следующим образом: Ctrl+K–B отмечает начало выделяемого блока, Ctrl+K–K — его конец, Ctrl+K–C — копирует, Ctrl+K–M — перемещает выделенный блок в позицию курсора и так далее.

Трехклавишные комбинации также не чувствительны к регистру. И работают также и при кириллической раскладке клавиатуры. В этом случае необходимо только нажимать вторую литерную клавишу вместе с той же клавишей Ctrl: то есть запись текущего файла потребует комбинации Ctrl+K + Ctrl+D, вызов нового файла — Ctrl+K + Ctrl+E и так далее.

В joe нет отдельной функции для переименования файла. Но при любой записи текущего документа следует запрос на подтверждение имени файла, которое может быть изменено. Следует только помнить, что дальнейшая работа происходит с исходным, а не переименованным файлом.

Кроме того, в joe доступны еще некоторые действия с файлами. Так, комбинация Ctrl+K–R вставляет текст из существующего файла в позицию курсора, Ctrl+K–K — записывает выделенный блок в виде нового файла (разумеется, запросив предварительно его имя). С помощью комбинации Ctrl+K–E можно открыть для редактирования другой существующий файл. При этом следует предложение ввести путь и имя, причем и для того, и для другого клавиша Tab выводит список файлов и подкаталогов текущего (на момент запуска joe) каталога.

Между одновременно открытыми файлами возможен обмен данными: или с помощью команд joe (то есть выделением блока в первом файле и его копированием или перемещением во второй), или с помощью мыши - стандартным выделением и вставкой щелчком ее средней кнопки. Второй способ, естественно, может применяться и для обмена между разными копиями joe, запущенными на отдельных виртуальных консолях.

Одновременно открытые файлы могут быть представлены как в полноэкранном виде, так и каждый в своем окне. Для переключения между режимами служит комбинация Ctrl+K–I. Размер каждого из выведенных окон может быть увеличен или уменьшен (Ctrl+K–G и Ctrl+K–T, соответственно), правда, только с шагом в одну экранную строку. Переключение между открытыми документами, вне зависимости от режима, осуществляется комбинациями Ctrl+K–N (вперед или вниз) и Ctrl+K–P (назад или вверх).

```
    I    Unnamed (Modified)   *SHELL* Row 22   Col 37  12:44  Ctrl-K H for help
Warning: no access to tty (Inappropriate ioctl for device).
Thus no job control in this shell.
/home/alv/work/articles/other/joe=>>ls -l
total 6777
drwxr-xr-x   3 alv  alv      512 12 апр 11:10 ./
drwxr-xr-x  10 alv  alv      512 12 апр 07:20 ../
-rwxr-xr-x   1 alv  alv    17731 12 апр 07:09 .joerc*
drwxr-xr-x   2 alv  alv      512 11 апр 10:57 .xvpics/
-rw-r--r--   1 alv  alv    24169 12 апр 11:28 joe.html
-rw-r--r--   1 alv  alv    23273 12 апр 11:05 joe.html~
-rwxr-xr-x   1 alv  alv   958477 12 апр 07:09 joe.tar.gz*
-rw-r--r--   1 alv  alv    21299 12 апр 09:58 joe1.html
-rw-r--r--   1 alv  alv      562  9 апр 16:31 joe_macros.txt
-rw-r--r--   1 alv  alv     9877 12 апр 11:02 ris01.png
-rw-r--r--   1 alv  alv  1270044 12 апр 11:01 ris01.tif
-rw-r--r--   1 alv  alv   557178 12 апр 11:01 ris02.png
-rw-r--r--   1 alv  alv  1404966 12 апр 11:00 ris02.tif
-rw-r--r--   1 alv  alv     8881 12 апр 10:56 ris03.png
-rw-r--r--   1 alv  alv  1267638 12 апр 10:56 ris03.tif
-rw-r--r--   1 alv  alv    11403 12 апр 10:56 ris04.png
-rw-r--r--   1 alv  alv  1275576 12 апр 10:57 ris04.tif
/home/alv/work/articles/other/joe=>>
```

Рис. 10.8: Окно командной среды внутри `joe`

Возможен в `joe` и независимый просмотр разных частей документа в отдельных окнах, для чего предназначена функция расщепления окна (Ctrl+K–O). Фрагменты из одной части файла могут быть перенесены в другую.

Универсальной комбинацией для окончания любой операции в `joe` является Ctrl+C. С ее помощью закрывается окно с текущим документом; если он был единственным в данном сеансе, одновременно происходит и выход из редактора. В обоих случаях следует запрос на сохранение выполненных изменений. Отказаться от выхода или закрытия файла можно повторным нажатием той же комбинации Ctrl+C. Она же используется для прекращения любой длящейся во времени (автоматическа проверка правописания, поиск) или требующей подтверждения операции.

Кроме этого, непосредственно из `joe`, без выхода, можно обращаться к командам оболочки, причем различными способами. Так, комбинация Ctrl+K–Z обеспечивает временный выход в оболочку, где можно вводить любые ее команды. А по завершении операций вернуться в `joe` можно командой `fg`. Единичную же команду из `joe` можно ввести, нажав Esc, а затем — ! (восклицательный знак).

Кроме этого, есть и более интересная возможность: открытие внутри `joe`, посредством комбинации Ctrl+K–' (апостроф), самостоятельного окна с командной оболочкой (рис. 10.8). Здесь можно выполнять любые команды с выводом их результатов на экран. После чего стандартной командой `exit` осуществляется выход из среды, а все результаты сохраняются обычным для `joe` образом в виде текстового файла.

Наконец, в `joe` обеспечивается ввод специальных символов. Так, нажатие клавиши ` (обратный апостроф) приводит к предложению ввести первый знак десятичного (0-9), шестнадцатиричного (`x`) или восьмеричного (`o`) кода символа. Если же вместо кода нажать Esc — можно ввести любую esc-

последовательность.

Таковы основные возможности joe для редактирования текстов общего характера.

Кроме этого, имеется ряд команд специального назначения, используемых при программировании (поиск ошибок, компиляция и проч.). Наконец, набор встроенных команд может быть почти неограниченно наращен с помощью макросов.

Для самостоятельного конструирования управляющих элементов joe предназначен внутренний язык макрокоманд. К сожалению, он не описан в экранной документации. Однако получить представление о его синтаксисе и возможностях можно, изучив внимательно конфигурационный файл joerc (о чем подробнее будет сказано ниже).

Кроме того, в joe существует режим протоколирования макрокоманд (см. рис. 10.6). Включается он комбинацией клавиш Ctrl+K–[(открывающая квадратная скобка), вслед за чем следует ввести номер макроса (от 0 до 9), выступающий как в качестве его имени, так и в роли запускающей клавиши.

Далее просто выполняются необходимые действия, после чего запись макроса останавливается комбинацией Ctrl+K–] (закрывающая квадратная скобка).

Для воспроизведения запротоколированного макроса используется комбинация Ctrl+K–цифра (где цифра — указанный при записи номер макрокоманды).

С помощью протоколирования макросов можно автоматизировать ввод наиболее нужных для конкретной задачи символов и их наборов, не предусмотренных штатным образом. Например, основных тэгов html для разметки web-страниц, таких, как параграф, разрыв строки, заголовки нескольких уровней, таблицы и списки. Запротоколированные в данном сеансе макросы могут быть помещены в тело существующего или нового документа (комбинацией клавиш Esc+ D) и в дальнейшем отредактированы в текстовом редакторе (том же joe, например).

За один сеанс можно запротоколировать не более 10 макросов (маркированных цифрами от 0 до 9). Более того, они будут действовать только в течении данного сеанса: по выходе из редактора записанные макросы сами собой не сохраняются. Однако есть средство сохранить их для дальнейшего использования, и при этом в неограниченном количестве. Для этого их следует поместить в соответствующую секцию конфигурационного файла .joerc.

Назначенные по умолчанию клавишные комбинации (включающие цифры от 0 до 9) не являются обязательными, их можно вручную заменить на любые другие, из числа не использованных ранее. После чего можно начать протоколирование команд сначала, повторно используя ту же нумерацию, снова

встроить их в joerc и так далее.

Таким образом можно легко автоматизировать процесс ввода тэгов HTML или XML, конструкций JavaScript, сценариев командной оболочки, разметки документов TEX, превратив joe в специализированный инструмент для решения почти любых задач.

Как уже говорилось, некоторые настройки joe можно выполнить интерактивно (вызвав их комбинацией Ctrl+T). Однако они весьма ограничены и к тому же будут иметь силу только в текущем сеансе. Более интересные и богатые возможности открываются при редактировании конфигурационного файла joerc, пример которого имеется в каталоге /etc/joe. Его следует скопировать в свой домашний каталог и переименовать в .joerc — именно этот файл ищется в первую очередь при загрузке редактора.

Файл .joerc разбит на четыре секции. Первая — это глобальные опции редактора, большая часть которых может быть задана также параметрами командной строки. Все они — односложные и имеют вид -имя_опции (установить данную опцию) или --имя_опции (отменить ее). Опция является установленной (или, напротив, отмененной), если ею начинается строка (это относится и ко всем остальным секциям .joerc). Если строка начинается с пробела или табуляции, дальнейшее ее содержание рассматривается как комментарий.

Комментарием является и все, отделенное пробелом от имени опции.

Остановимся на некоторых ключевых опциях первой секции. Обязательно следует включить (то есть удалить пробел в начале строки, если он имеет место быть) опцию -asis. Это необходимо для правильного отображения символов кириллицы — иначе они могут показаны латинской транслитерацией. Полезна также установка опций -lightoff - выключение подсветки выделенного блока после его перемещения или копирования, -marking — подсветка текста между началом выделяемого блока и текущей позицией курсора. Можно отменить также создание страховых копий или, напротив, определить место для их помещения, отличное от исходных файлов (опции --nobackups и -backpath path соответственно).

Есть возможность задания количество строк и колонок (знаков в строке) на экране, отличное от заданных для текущего терминала (виртуальной консоли) в целом, что задается опциями -lines 999 и -columns 999, где 999 — количество строк и знаков соответственно.

В этой же секции настраивается вид статусной строки (вывод которой можно отключить опцией -nosta). Он определяется двумя опциями -lmsg и -rmsg.

Первая указывает компоненты, выводимые в левой части строки, вторая — в правой. В любой из них можно вывести индикацию режимов (забивки или вставки, переноса слов, автоотступов), имя файла и показатель его изменения, текущее положение курсора (в строках, колонках или знаках) и т.д. Пер-

вый знак левой части статусной строки — escape-последовательность, определяющая ее общий вид: инвертирование цветов, выделение подчеркнутым или полужирным начертанием, мерцание.

Вторая секция конфигурационного файла — это локальные опции, которые можно определить отдельно для файлов различных типов. Для этого в ее составе создаются субсекции вида

- * — все файлы,
- *.html — документы HTML,
- *.c — программы на языке C,
- *rc — конфигурационные файлы

и так далее. Напомним, что знак маски (*) должен обязательно начинаться с начала строки.

Для каждой субсекции можно независимо задать такие параметры, как перенос слов (или его отключение), автоматический отступ, величина табуляции и т.д., а также назначить собственную раскладку клавиатуры, отличную от определенной для консоли в целом. Кроме того, с файлами любого типа можно связать макросы, выполняемые при их загрузке или записи.

Третья секция описывает вид экранов помощи, выводимых клавишной комбинацией Ctrl+K-H (см. приложение). Экраны эти могут быть изменены как с позиций внешнего вида (инверсия цветов, выделение или подчеркивание и т.д.), так и по существу. В частности, здесь можно задать специальные экраны помощи для собственных макрокоманд. Более того, редактированием этой секции можно выводить и подсказки на русском (или каком-либо еще) языке.

Четвертая секция — разного рода ключевые последовательности, или связки (key bindings), в том числе и макрокоманды. Они могут быть быть определены отдельно для всех окон (:windows), окна редактируемого текста (:main), и так далее.

Рассмотрение секции показывает, что все клавишные комбинации joe представляют собой макрокоманды, именно здесь и определенные. Поэтому, во-первых, имеется возможность переопределения клавиатурных комбинаций, назначенных для штатных команд joe по умолчанию. Во-вторых, штатные команды joe могут быть пользовательскими макрокомандами. Для чего достаточно их запротоколировать и встроить в какой-либо раздел (например, :windows или :main) четвертой секции файла .joerc, одновременно изменив их имена и закрепленные клавишные комбинации.

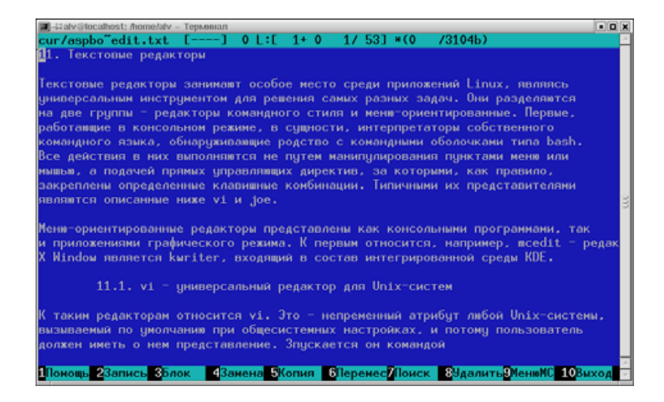


Рис. 10.9: Текстовый редактор mcedit — вид по умолчанию

10.3 Меню-ориентированный редактор mcedit

Типичным примером консольных меню-ориентированных текстовых редакторов можно считать `mcedit`. Он встроен в файловый менеджер `Midnight Commander`, но может использоваться и как самостоятельное приложение.

Вызывается `mcedit` по клавише `F4`, зафиксированной на текстовом файле. Но его можно запустить и автономно, просто набрав `mcedit` в командной строке, с указанием имени файла или без оного (в последнем случае открывается пустой файл).

При запуске `mcedit` обнаруживается рабочее поле, обрамленное сверху статусной строкой, а снизу панелью функциональных клавиш. Кроме того, большинство действий, доступных через меню, также дублируется комбинациями горячих клавиш.

Строка меню активизируется нажатием клавиши `F10` или щелчком мыши на статусной строке. `mcedit` при включенном сервере `gpm` поддерживает указательно-позиционирующие функции мыши. Пункты меню — следующие (рис. 10.10):

- Файл, где объединены действия по открытию и созданию файлов, их сохранению (в том числе и под другим именем), вставке существующего файла в текущий и, напротив, записи текущего файла в существующий; здесь же — выход и вызов;
- Меню пользователя, доступно только в том случае, если mcedit вызван из MC;
- Редактирование, где предусмотрены выделение блока и столбца (то есть

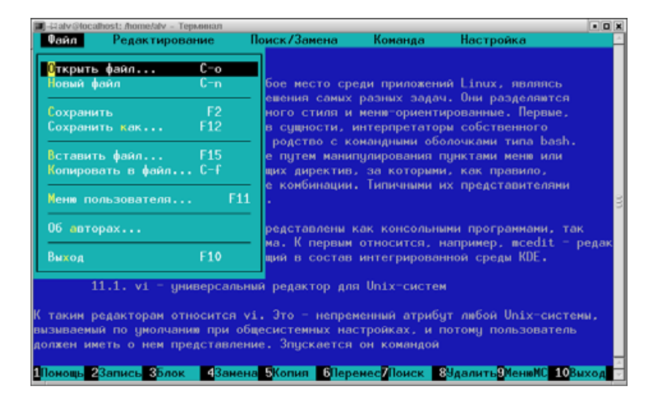


Рис. 10.10: Строка меню mcedit

вертикального блока), их копирование, удаление и перемещение, отмена операций, переход в начало и конец файла, а также переключение режима вставки/замены при наборе и редактировании текста;

- Поиск/замена, включая использование регулярных выражений, чувствительность к регистру, обратный поиск, глобальную замену и т.д. (рис. 10.11);
- Команда;
- Настройка.

На двух последних пунктах следует остановиться подробнее. Через меню Команда выполняются такие действия, как переход к строке (по ее номеру) и к парной скобке любого вида (открывающей или закрывающей), вставка символа, даты и времени, сортировка в выделенном блоке, форматирование абзаца (в соответствие с правилами переноса слов, установленными в меню Настройка), проверка орфографии и даже отправка электронной почты.

Кроме того, в меню *«Команда»* предусмотрено создание макросов путем протоколирования действий. Для этого нажатием комбинации `Ctrl`+`R` следует перейти в режим записи, выполнить необходимые действия, повторным нажатием `Ctrl`+`R` выйти из режима записи и в появившейся панели приписать созданной последовательности любой символ с клавиатуры. Далее исполнение макроса возможно через меню *«Команда»*- *«Выполнить макрос»* или нажатие комбинации клавиш `Ctrl`+`A`, а затем — присвоенной макросу клавиши. Записанные макросы сохраняются в файле `~/.cedit/cooledit.macros`, где могут быть отредактированы вручную.

В меню *«Настройка»* два пункта — *«Разное»* и *«Режим сохранения»*

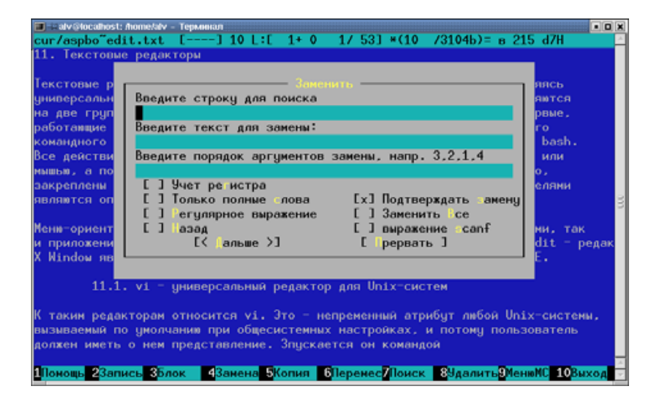


Рис. 10.11: Панель поиска mcedit

(рис. 10.13). В первом (рис. 10.14) можно установить такие опции редактора, как эмуляция раскладки клавиш, подтверждение на запись файла, цветовыделение синтаксических конструкция языков программирования и разметки и т.д.

Здесь же устанавливается режим переноса слов. По умолчанию он выключен, но может быть включена динамическая разбивка абзаца (по достижении границы экрана) или автоматический перенос слов после заданного (в поле Позиция переноса строк) количества символов. Именно в соответствии с этими правилами осуществляется форматирование абзацев (через меню Команда -> Форматировать абзац).

Режим сохранения можно определить в одном из двух вариантов (рис. 10.15):

- Быстрое сохранение (с кэшированием дисковых операций),
- Безопасное сохранение (немедленная запись файла на диск).

Можно также включить или отключить создание резервных копий при записи файла.

В mcedit предусмотрено разнообразное выделение текстовых фрагментов: по клавише `F3`, мышью, клавишами управления курсором при нажатом `Shift`. Если при копировании и перемещении предназначенный для этого фрагмент выделен мышью, курсор в новую позицию можно переместить только стрелками, иначе выделение пропадет. Удаление выделенного фрагмента возможно только клавишей `F8`, но не `Del`. Наконец, вставка выделенного мышью фрагмента щелчком средней кнопки осуществляется при нажатой клавише `Shift` или `Ctrl`.

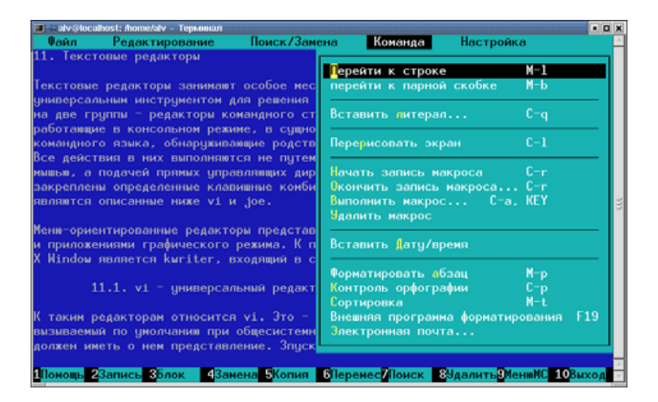


Рис. 10.12: Меню *«Команда»*

mcedit почти не имеет ограничений на размер открываемого файла — оно составляет 16 Мбайт.

В целом mcedit представляет собой идеальный редактор для эпизодического применения, например, правки конфигурационных файлов. А удобные и привычные пользователю средства навигации и редактирования вместе с возможностью протоколирования макросов делают его пригодным и для повседневной работы с текстами любого характера и объема.

10.4 Текстовый редактор kwrite для среды *KDE*

Среда *KDE* располагает двумя штатными средствами для редактирования текстов — kedit (в русскоязычной версии именуемый **Текстовым Редактором** или **Простым Редактором**) и kwrite (**Улучшенный редактор**). Первый — достаточно простой инструмент для начинающего, тогда как kwrite — мощный и развитый текстовый редактор универсального назначения.

Редактор kwrite запускается из стартового меню *«***К***»- «Редакторы»- «Улучшенный редактор»* или из Панели, где он представлен среди приложений по умолчанию. Кроме того, он может быть запущен одноименной командой в строке эмулятора терминала или минитерминала.

При запуске kwrite открывается окно с рабочим полем, строкой меню и панелью инструментов вверху и статусной строкой внизу (рис. 10.16).

Через панель инструментов (в конфигурации по умолчанию, поскольку она настраиваема) доступны лишь наиболее обычные манипуляции, как то: создание нового файла или открытие существующего, запись, печать, отмена и повтор,

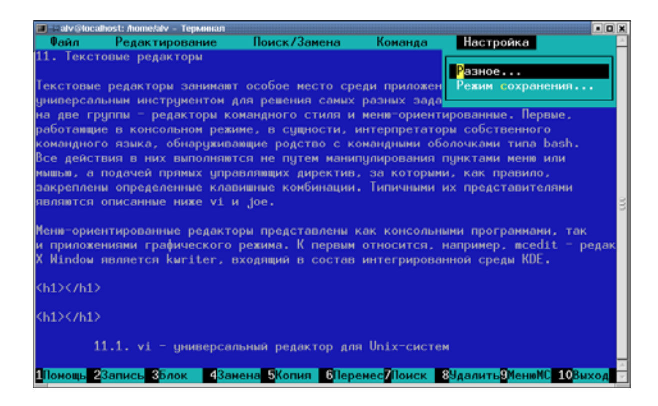


Рис. 10.13: Меню Настройка

вырезание, копирование, вставка, а также поиск.

Возможности, доступные через меню, много шире. Оно включает пункты:

- Файл,
- Правка,
- Перейти,
- Инструменты,
- Настройки,
- Помощь.

В каждом из них сгруппированы действия соответствующей направленности. Большинство доступных через меню действий дублируется комбинациями горячих клавиш.

Через меню *«Файл»* (рис. 10.17) осуществляется создание (Ctrl+N) и открытие документов (Ctrl+O), в том числе из списка недавних, их сохранение (Ctrl+S) и переименование, печать (Ctrl+P) и выход из редактора (Ctrl+Q). Здесь можно также открыть *«Новое окно»* в текущем сеансе редактора и создать *«Новый вид»*. В первом случае открывается просто пустое окно с параметрами по умолчанию, во втором — создается копия текущего окна с редактируемым в исходном окне документом.

Меню *«Правка»* (рис. 10.18) начинается с пунктов *«Отменить»* действие (Undo, Ctrl+Z) и *«Повтор»* (Redo, Ctrl+Shift+Z), оба с расшифровкой последних манипуляций. Кроме того, имеется *«Журнал отмен/повторов»*, позволяющий выполнить то или другое сразу для группы

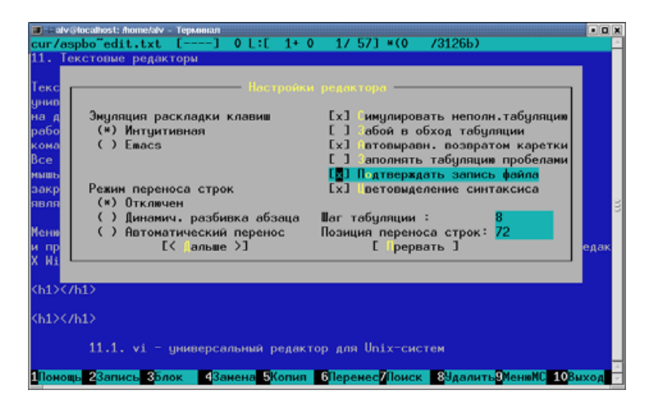


Рис. 10.14: Настройка основных опций mcedit

операций (рис. 10.19). Уровней отмены/возврата по умолчанию 50, но число это настраиваемо.

Далее следуют стандартные операции *«Вырезать»* (Ctrl+X), *«Копировать»* (Ctrl+C), *«Вставить»* (Ctrl+V), *«Вставить файл»*, и операции выделения: *«Выбрать все»* (Ctrl+A), *«Отменить все»* (выделение), *«Инверсия выделения»*.

В этом же меню доступны операции поиска и замены: *«Поиск»* (Ctrl+F), *«Продолжить поиск»* (F3), *«Заменить»* (Ctrl+R). В числе опций поиска и замены (рис. 10.20) — использование регулярных выражений, различие регистров, поиск/замена в выбранном фрагменте и т.д.

Меню *«Перейти»* (рис. 10.21) позволяет перейти на строку по ее номеру, *«Добавить метку»* в текущей позиции курсора с автоматической ее нумерацией (Ctrl+M) и *«Установить метку»* с произвольным номером (рис. 10.22). В обоих случаях список меток добавляется к нижней части меню (см. рис. 10.21).

Меню *«Инструменты»* включает пункты (рис. 10.23):

- Проверка правописания, осуществляемая подключаемой внешней программой, например, aspell или ispell;
- Установить и снять абзацный отступ (Ctrl+L и Ctrl+U для текущей строки, соответственно), а также убрать все отступы в документе;
- *«Закомментировать»* и *«Разкомментировать»* текущую строку, по умолчанию знак комментария — пробел в начале строки.

Пункт *«Настройки»* позволяет изменить многочисленные опции редактора,

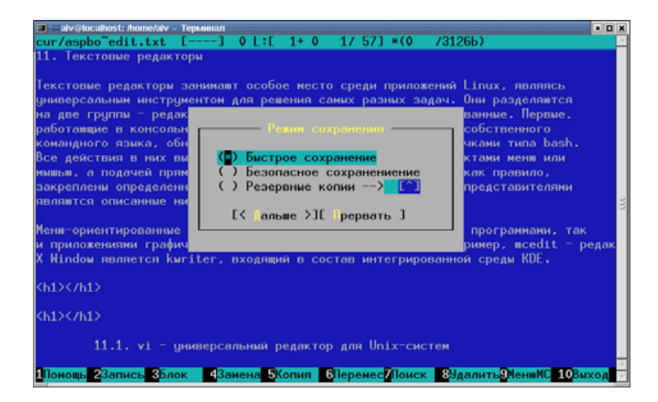


Рис. 10.15:

например, показать или скрыть панель инструментов, строку состояния и путь к редактируемому файлу в заголовке окна (рис. 10.24).

С помощью пункта *«Настроить привязки клавиш »*(рис. 10.25) можно переопределить комбинации горячих клавиш по умолчанию для предусмотренных исходно действий, а также приписать им собственные клавишные комбинации из допустимых комбинаций — `Shift`, `Ctrl`, `Alt` и алфавитно-цифровой символ в любых сочетаниях.

Как уже говорилось, поддается настройке и панель инструментов (рис. 10.26) — можно добавить в нее такие клавиши, как проверка орфографии или продолжение поиска, а также выпадающие списки недавно открывавшихся файлов или выбора символа конца строки, в результате чего панель может принять вид вроде приведенного на рис. 10.27. Можно изменить порядок клавиш и списков или удалить отдельные элементы панели.

Основные опции настройки `kwrite` сгруппированы в одноименном пункте меню *«Настройка»* (рис. 10.28). В первом ее пункте *«Цвета»* можно настроить цвет текста и фона, выделенного фрагмента, найденного текста.

Далее, поддаются настройке отступы (рис. 10.29). Можно включить автоматический ввод абзацных отступов, замену отступов пробелами, включить/выключить сохранение лишних пробелов, вводимых клавишей `Пробел` и т.д.

Опции выделения (рис. 10.30) включают: устойчивые выделения, сохраняющиеся после изменения положения курсора, или, напротив, разовые выделения, как это принято в большинстве программ для X Window System, включение автокопирования мышью (то есть вставки по щелчку средней ее кнопкой) и вертикального выделения отдельными колонками или их группами.

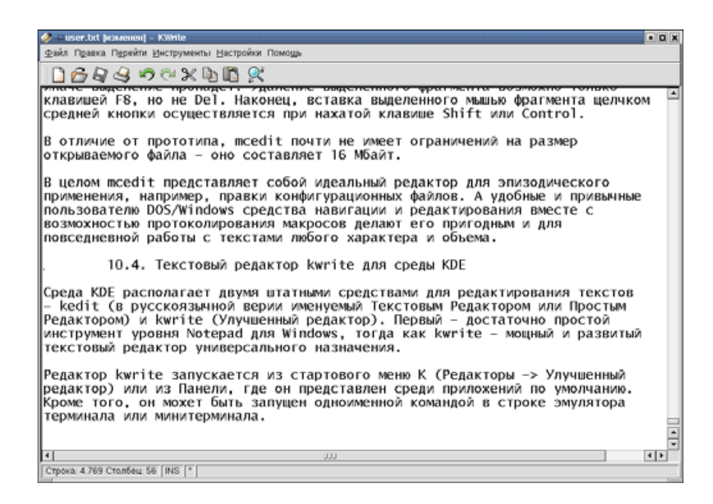


Рис. 10.16: Текстовый редактор `kwrite`

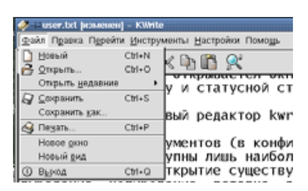


Рис. 10.17: Меню *«Файл»* редактора `kwrite`

В настройках редактирования переключателями устанавливаются (рис. 10.31):

- автоматический перенос слов,
- удаление завершающих строку пробелов,
- автоматический ввод парной скобки,
- поведение клавиш `Home` и `PageUp`/`PageDown`, то есть опции, однозначно определяемые в Windows -программах, но варьирующиеся в приложениях для Linux. Здесь же определяются граница переноса слов и величина табуляции (в символах), а также число уровней отмены/возврата операций.

В пункте проверки орфографии (рис. 10.32) можно выбрать внешнюю программу этого назначения (`ispell`), словарь для языка — русского или английского в английском и американском вариантах, а также кодировку симво-

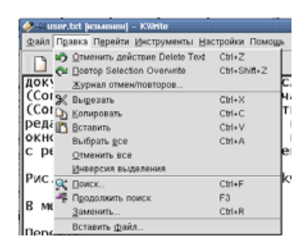


Рис. 10.18: Меню Правка редактора kwrite

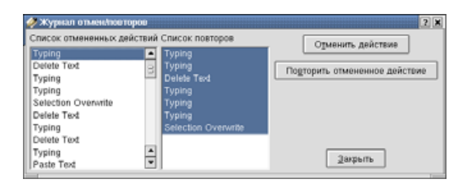


Рис. 10.19: Журнал отмен/повторов

лов. Переключателями устанавливаются такие опции, как включение в словарь флективных форм и различение написанных слитно слов.

Среди прочих настроек важен пункт *«Настройка подсветки»*. Во-первых, именно здесь, в закладке *«По умолчанию»* (рис. 10.33) определяются гарнитура шрифта (из всех доступных в X Window System, как пропорциональных, так и моноширинных), его кегль, начертание и цвет, а также набор символов. Причем все они могут быть определены независимо для разных элементов текста ключевых слов, десятичных чисел, комментариев и т.д. (рис. 10.34).

Во-вторых, в закладке Режимы подсветки (рис. 10.35) шрифтовое оформление может быть установлено для различных языковых режимов (C, C++, Java, HTML и т.д., рис. 10.36). Подсветка для каждого режима может включаться автоматически при загрузке файла с фиксированным расширением (напри-

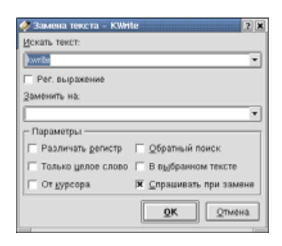


Рис. 10.20: Опции поиска и замены

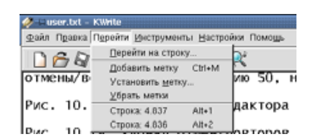


Рис. 10.21: Меню Перейти редактора kwrite

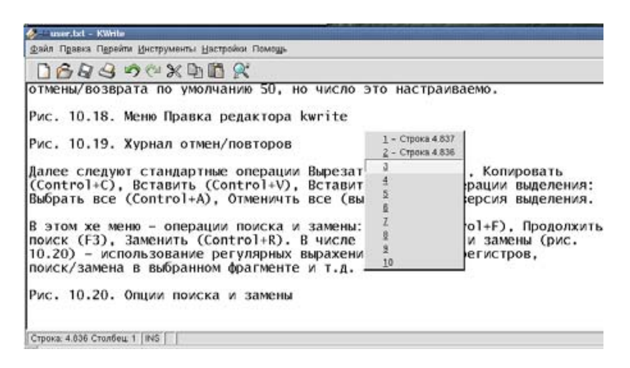


Рис. 10.22: Установка метки с произвольным номером

мер, `*.html`) или MIME-типом (`text/html`) и конструкций, характерных для каждого из языков.

Кроме того, независимая подсветка может быть настроена для отдельных конструкций каждого из поддерживаемых языков. Например, для режима HTML независимо могут быть настроены характеристики шрифтов для обычного текста, тэгов и его аргументов, переменных и комментариев.

Далее, в меню *«Настройки»* можно включить разовое выделение вертикальных блоков, что аналогично включению соответствующей опции в пункте *«Настройки»*- *«Настроить KWrite»*- *«Выделение»* (см. рис. 10.30).

Последние два пункта позволяют выбрать, во-первых языковый режим для данного документа, активизирующий соответствующие опции подсветки (рис. 10.36) и символ конца строки (см. рис. 10.24) в формате UNIX (LF), DOS/Window (LF CR) или Macintosh (CR).

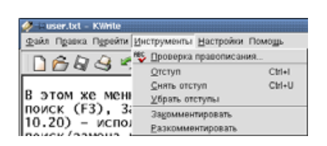


Рис. 10.23:

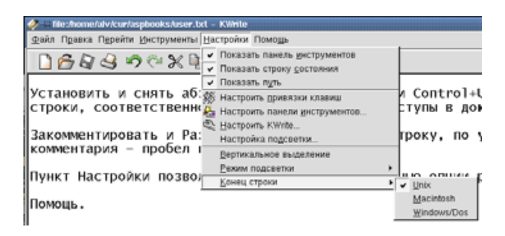


Рис. 10.24: Меню Настройки редактора kwrite

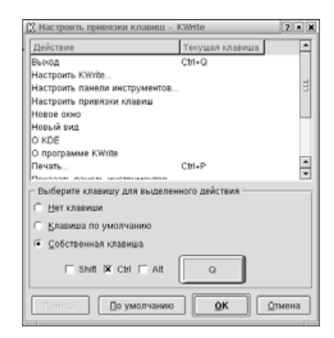


Рис. 10.25: Настройка привязки клавиш

Наконец, в пункте *«Помощь»* вызывается соответствующий раздел руководства *KDE* в формате HTML (также и клавишей `F1`). Возможно также получение контекстной справки (`Shift`+`F1`) общим для всех *KDE*-приложений методом — указанием курсором, принявшим форму вопросительного знака, на какой-либо элемент интерфейса.

10.5 Конверторы кириллических кодировок

Проблема преобразования русскоязычных текстов из одной кодировки (набора символов) кириллицы в другой тесно связана с обработкой текстов как таковых.

Как известно, в настоящее время распространено не менее пяти кодировок кириллицы:

- KOI8, принятая в Linux и большинстве других UNIX-систем,
- кодировка MS DOS (она же CP866, или альтернативная, используемая также в OS/2),

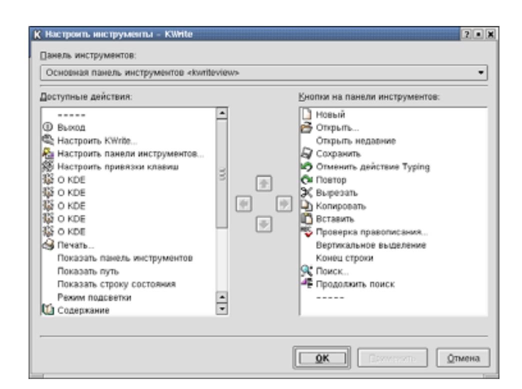


Рис. 10.26: Настройка панели инструментов

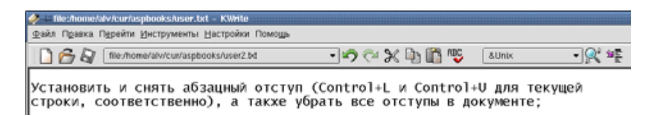


Рис. 10.27: Панель инструментов с добавленными клавишами и выпадающими списками

- кодировка Windows , или CP1251 (рекомендуемая к использованию),
- ISO8859-5, она же кодировка ГОСТ, или основная; вопреки названию, применяется редко — только на рабочих станциях Sun с ОС Solaris,
- кодировка MacOS.

В принципе, Linux (и многие его приложения) может работать с любой кодировкой, в том числе и с CP1251, пользующейся наибольшим распространением и имеющей ряд преимуществ над другими кириллическим кодировками, основные из которых — это поддержка нескольких языков, основанных на кириллице (русский, украинский, болгарский, беларусский). Однако нет оснований отказываться от KOI8 особенно в случаях удаленного администрирования систем, не имеющих поддержки CP1251. Заметьте, что процесс перекодирования текстовых файлов совсем не сложен.

Для этого предназначен ряд программ, из которых в дистрибутив **ASPLinux** включена `iconv`, которая стандартно входит в пакет `glibc`. Формат использования этой утилиты:

```
iconv -f [исходная_кодировка] -t [целевая_кодировка] old_file > new_file
```

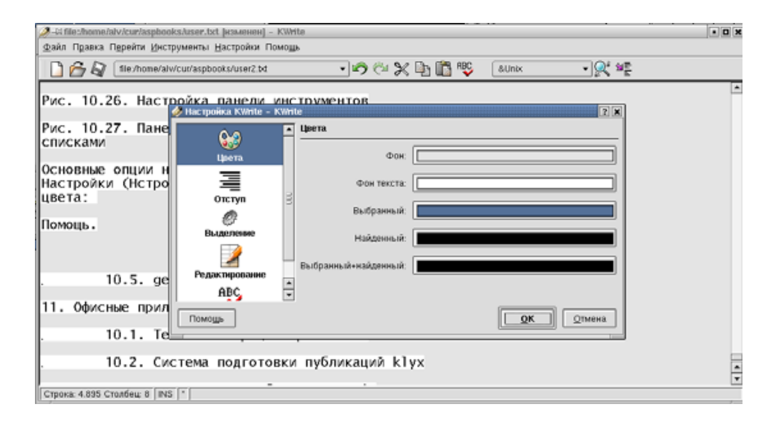


Рис. 10.28: Панель Настройки редактора kwrite

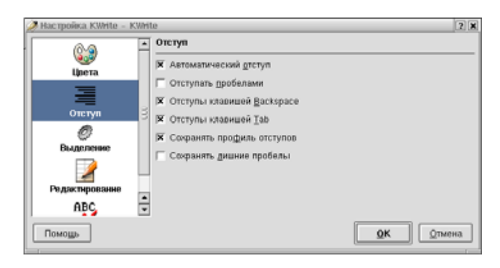


Рис. 10.29: Настройка отступов

где следует обратить внимание на символ перенаправления вывода в новый файл.

Допустимые значения для опций `-f` и `-t` можно определить при помощи опции `--list`. В их числе, кроме KOI8-R (и KOI8-U) — CP866, CP1251, MAC, ISO8859-5. Большинство наборов символов могут указываться различными способами. Так, для кодировки DOS допустимые значения опций `-f` и `-t` могут указываться как CP866, IBM866 и даже просто 866, для кодировки Windows — как CP1251 или WINDOWS-1251 и т.д.

Программа `iconv` очень проста в использовании, но способна преобразовывать только один файл. При необходимости большого количества перекодируемых документов (например, преобразовании целого сайта со всеми входящими в него html-файлами) можно использовать программу rusconv. Она не входит в состав дистрибутива, но может быть получена с сайта `http://beta.math.spbu.ru/~prof/w_re/`, где содержится также подробная документация по ее применению (на русском языке).

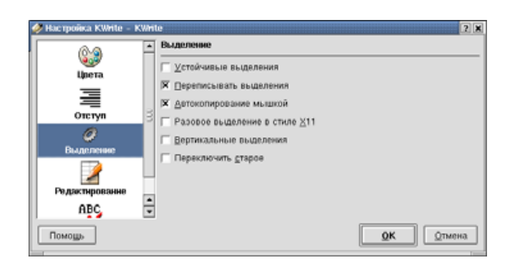


Рис. 10.30: Настройка опций выделения

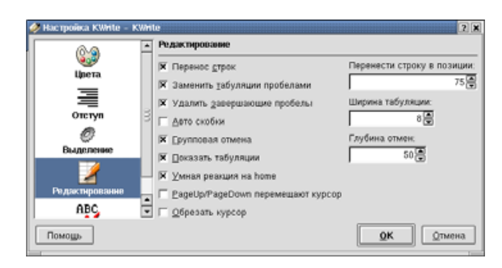


Рис. 10.31: Настройка опций редактирования

Преимущество программы `rusconv` — возможность перекодирования нескольких (или даже сразу всех) файлов каталога. При этом, если не указать новый целевой каталог, перекодированные файлы будут помещены в исходный каталог с изменившимся расширением, соответствующим новой кодировке (например, для KOI8 — `*.koi`).

Второе достоинство программы `rusconv` — то, что она существует в версиях для всех UNIX-систем, а также для MS DOS и Windows.

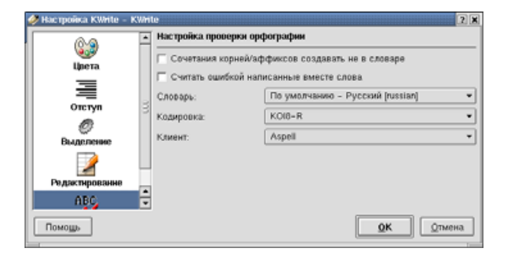


Рис. 10.32: Настройка проверки орфографии

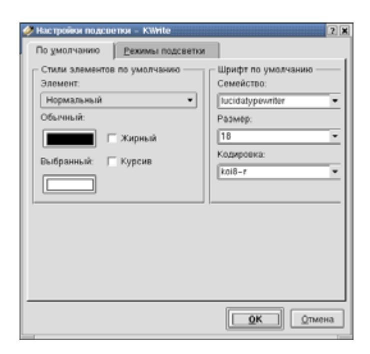


Рис. 10.33: Настройка шрифтового оформления рабочей области редактора `kwrite`

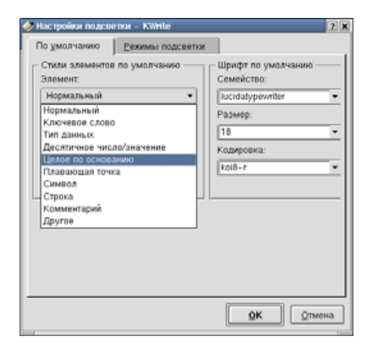


Рис. 10.34: Элементы текста, для которых возможно независимое шрифтовое оформление

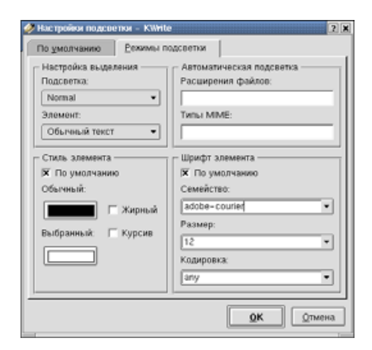


Рис. 10.35: Настройка режимов подсветки

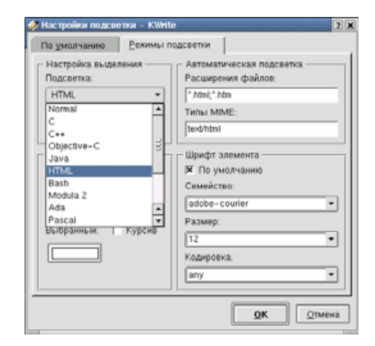


Рис. 10.36: Языковые режимы, для которых возможна настройка подсветки

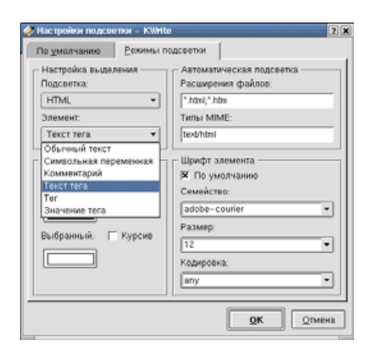


Рис. 10.37: Настройка подсветки для отдельных конструкций языкового режима

Глава 11

Офисные приложения

До недавнего времени как количество, так и качество офисных приложений (то есть текстовых процессоров, электронных таблиц и т.д.) под Linux оставляло желать лучшего. Однако ныне положение изменяется. И пользователь **ASPLinux** имеет возможность выбора между полнофункциональными интегрированными пакетами офисного назначения и менее ресурсоемкими монофункциональными программами. Главные из них, текстовые процессоры, представлены в дистрибутиве **ASPLinux** современным офисным пакетом **OpenOffice**, классическим процессором **AbiWord** и интегрированным пакетом **KOffice**.

Во избежание недоразумений следует отметить, что в переводной литературе по Linux под текстовыми процессорами (text processor в англоязычных источниках) понимаются не WYSIWIG-системы[1] обработки текста (типичный пример — всем известный MS Word или отечественный Lexicon), а потоковые (не интерактивные) программы форматирования текста, в сущности, интерпретаторы какого-либо языка разметки (именно к этому классу относятся программы groff и $\TeX$).

Собственно же визуальные системы для работы с текстами именуются процессорами слов (word processor). Об этом различии с устоявшейся русскоязычной терминологией (применяемой, в частности, для Windows-приложений), следует помнить при знакомстве с дополнительными источниками информации.

Однако стоит заметить, что текстовые процессоры — главный, но не единственный компонент офисных приложений. Почти столь же важны для пользователя электронные таблицы, программы подготовки деловой графики и презентаций и т.д. В настоящее время все это обычно объединяется в т.н. интегрированные офисные пакеты.

Наиболее развитым и мощным из таких пакетов, отвечающим всем требованиям современного делопроизводства, для Linux является **OpenOffice**, входя-

[1] WYSIWIG (What You See Is What You Get) — Вы получите то, что видите.

щий в дистрибутив **ASPLinux**. Документация для этого пакета слишком обширна, чтобы освещать ее в рамках настоящего руководства. Исчерпывающую информацию по всем вопросам, связанным с **OpenOffice**, можно найти на сайте http://www.openoffice.org/.

11.1 Краткое введение в OpenOffice

OpenOffice (полное название — **OpenOffice.org**) — это полноценный офисный пакет программ. Он включает в себя:

- Текстовый редактор и текстовый процессор **Writer**;
- Редактор HTML файлов, т.е. WWW-страниц;
- Систему электронных таблиц **Calc**;
- Систему подготовки презентаций **Impress**;
- Редактор рисунков **Draw**;
- Редактор формул **Math**.

OpenOffice во многом похож на другие, давно известные и привычные офисные наборы программ. Однако есть у него и важные отличия:

- Существуют версии **OpenOffice** и под операционную систему Linux, и под Windows (поддерживаются также другие операционные системы — FreeBSD, Mac OS X). Это позволяет полноценно работать с одними и теми же документами на компьютерах с разными ОС.
- **OpenOffice** легально свободно распространяется с исходными текстами. Таким образом, отсутствует проблема нелицензионного программного обеспечения.
- Форматы файлов **OpenOffice** открыты, документированы и широко известны. Кроме того, **OpenOffice** может успешно работать с файлами многих форматов, включая и файлы, созданные другими известными офисными пакетами.

В этом кратком введении мы, разумеется, не сможем описать все богатые возможности **OpenOffice.** Мы приведём руководство лишь по установке системы, а также основным действиям в текстовом процессоре и редакторе HTML файлов. Информация о работе остальной части **OpenOffice** доступна в системе подсказки (правда, на английском языке).

Во многом работа в **OpenOffice** схожа с работой в любом другом офисном пакете. При этом **OpenOffice** в состоянии работать с файлами данных многих

распространённых форматов. Поэтому мы надеемся, что начало применения **OpenOffice** не вызовет у Вас особых затруднений.

11.1.1 Запуск OpenOffice

Запуск компонентов **OpenOffice** производится из меню **OpenOffice** в KDE или GNOME:

- Текстовый процессор **Writer** запускается пунктом *«Текстовый документ»*;
- Редактор HTML запускается пунктом *«Документ HTML»*;
- Система электронных таблиц **Calc** запускается пунктом *«Документ электронной таблицы»*;
- Система подготовки презентаций **Impress** запускается пунктом *«Презентация»*;
- Редактор рисунков **Draw** запускается пунктом *«Рисунок»*;
- Редактор формул **Math** запускается пунктом *«Формула»*.

Непосредственно после первого запуска следует произвести некоторые дополнительные настройки для корректной работы с русским языком.

11.1.2 Установка и начальная настройка OpenOffice

Первоначальная установка пакета **OpenOffice** производится при установке **ASPLinux**.

Запустить любой из компонентов **OpenOffice** можно прямо из панели инструментов, нажав на соответствующую иконку. Мы рекомендуем начать с **Writer** (можно также выбрать подпункт меню *«OpenOffice»* KDE или GNOME *«Текстовый документ»*). При запуске появится окно, предлагающее произвести импорт адресной книги. Нажмите кнопку **«Отменить»**.

Для корректной работы **OpenOffice** с русским языком, включая проверку орфографии и редактирование HTML-файлов, следует произвести некоторые дополнительные настройки. Чтобы сделать это, выберите в главном меню пункт *«Сервис»*, затем *«Параметры»*. Откроется окно настройки **OpenOffice** (рис. 11.1).

В этом окне можно изменять очень многие настройки, касающиеся работы различных компонентов **OpenOffice**. Однако мы опишем лишь установки, которые необходимо произвести для правильной работы системы с русским языком.

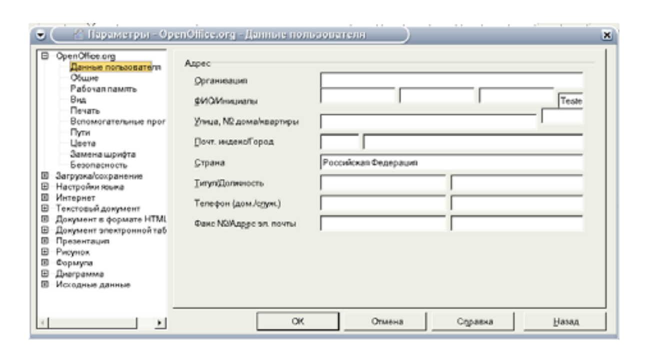


Рис. 11.1: Окно настройки **OpenOffice**

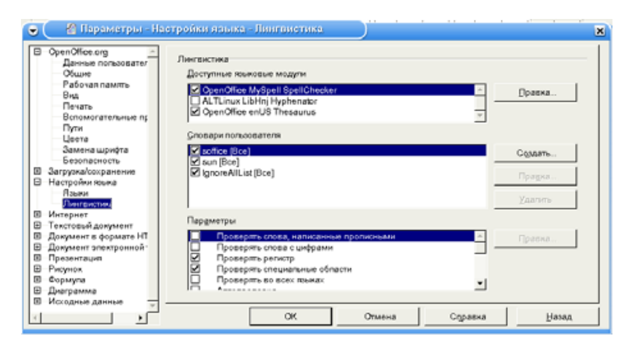


Рис. 11.2: Окно настройки языка, пункт *«Лингвистика»*

В левой части окна выберите пункт *«Настройки языка»*. Раскроется дерево подпунктов. Выберите *«Лингвистика»* (рис. 11.2).

В правой верхней части окна должен быть выбран пункт *«OpenOffice MySpell SpellChecker»*, и рядом с ним должна быть включена «галочка». Если это не так — включите «галочку» щелчком мыши. Нажмите на кнопку **«Правка. . . »** в правой верхней части окна. Появится окно **«Правка модулей»** (рис. 11.3).

В поле язык должно быть выбрано *«Русский»*; если это не так, произведите выбор щелчком мыши. Затем включите «галочку» перед пунктом *«OpenOffice MySpell SpellChecker»*.

Кроме русского, Вы можете таким же образом включить проверку орфографии для украинского, немецкого и французского языков. Для английского языка она включена изначально. Включив проверку орфографии для нужных Вам языков, нажмите кнопку **«Закрыть»**. Окно **«Правка модулей»** закроется. Вы вернетесь к установке параметров **OpenOffice**. Выберите пункт

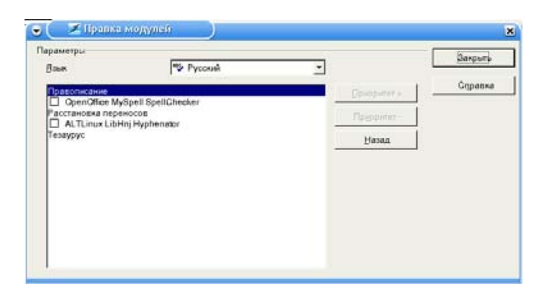


Рис. 11.3: Окно настройки языка, пункт *«Правка модулей»*

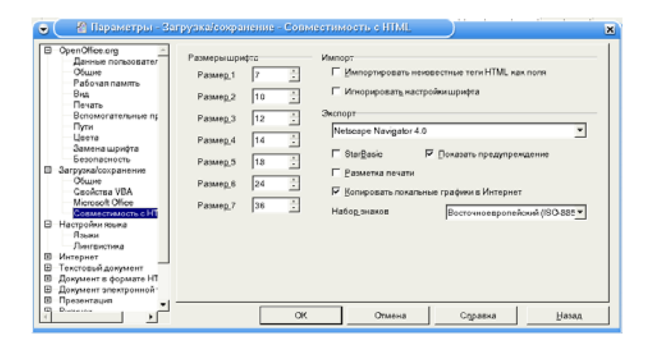


Рис. 11.4: Окно настройки языка, пункт *«Совместимость с HTML»*

«Загрузка/сохранение» в левой части окна. Раскроется дерево подпунктов. Выберите *«Совместимость с HTML»* (рис. 11.4).

В поле *«Набор знаков»* следует выбрать кодировку, в которой нужно сохранять HTML-файлы.

Стандартная кодировка для Интернета — *«Кириллический (KOI8-R)»*. Выберите пункт *«Текстовый документ»* в левой части окна. Раскроется дерево подпунктов. Выберите *«Основные шрифты»* (рис. 11.5).

Здесь нужно выбрать шрифты, используемые для различных видов текста. Необходимо указать шрифты, установленные в системе и корректно поддерживающие русский язык — например, Helvetica. Завершив установку параметров, нажмите кнопку **«OK»** в нижней части окна. Теперь Вы можете работать с офисным пакетом программ **OpenOffice**.

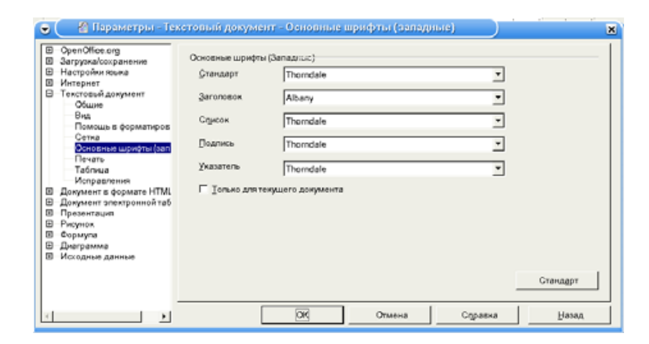


Рис. 11.5: Окно настройки языка, пункт *«Основные шрифты»*

11.1.3 Текстовый редактор/процессор Writer

Базовые функции редактирования текста

Работа с базовыми функциями **Writer** — ввод текста, перемещение по тексту, выделение блока, его вырезание, копирование или вставка — практически аналогична работе в любом из известных текстовых процессоров с графическим интерфейсом (например, для Windows или Macintosh).

Режимы просмотра В **OpenOffice Writer** предусмотрено два режима просмотра и редактирования текста на экране — *«разметка страницы»* и *«разметка online»*. При использовании разметки страницы на экране показывается страница в таком же виде, в каком она должна быть распечатана. Если же установлена *«разметка online»*, текст показывается без разбиения на страницы и с шириной во всё окно **Writer** — т.е. так, как текст обычно показывается в веб-браузерах. Режим *«разметка online»* удобен при подготовке документов распространяемых в электронном виде, а также при написании текстов большого объёма. Переключение между режимами *«разметка страницы»* и *«разметка online»* производится в главном меню — пункт *«Вид»*, затем *«Разметка online»*. Кроме того, на панели, расположенной слева от текста, имеется иконка, позволяющая переключать режим просмотра.

Масштаб **Writer** позволяет установить масштаб (zoom) просмотра текста в процентах. Чтобы изменить масштаб, следует выбрать в главном меню пункт *«Вид»*, затем *«Масштаб»*. В появившемся окне (рис. 11.6) можно выбрать из нескольких фиксированных вариантов масштаба либо, выбрав пункт *«Плавное»*, указать произвольное значение масштаба.

Затем следует нажать кнопку **«OK»**. Новый масштаб будет установлен. Ок-

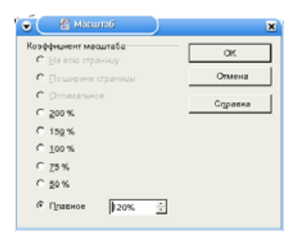


Рис. 11.6: Изменение масштаба документа

но выбора масштаба можно также вызвать, дважды щёлкнув левой кнопкой мыши на значение масштаба, которое показано в строке состояния в нижней части окна **Writer**.

Проверка орфографии **Writer** позволяет производить проверку орфографии как во время набора текста (подчёркивание слов с ошибками), так и при вызове функции проверки. Чтобы включить или выключить подчёркивание слов с ошибками, выберите в главном меню пункт *«Сервис»*, затем *«Правописание»*, *«Автопроверка»*.

Когда слово подчёркнуто, Вы можете выбрать из предлагаемых системой правильных вариантов или добавить данное слово в словарь. Для этого щёлкните по слову правой кнопкой мыши. Появится меню, в котором приведены правильные варианты, а также имеется пункт *«Добавить»*. Чтобы проверить орфографию во всём тексте, выберите в главном меню пункт *«Сервис»*, затем *«Правописание»*, *«Проверка»*, либо нажмите клавишу F7. Начнётся процесс проверки.

При обнаружении слова, отсутствующего в словаре, появляется окно **«Правописание»**.

В данном окне Вы можете указать, следует ли исправить данное слово (можно выбрать один из предложенных системой вариантов или ввести свой), заменять его на выбранный вариант во всём тексте, пропустить его (оставив без изменений) или пропускать во всём тексте. Кнопка **«Добавить»** позволяет добавить данное слово в словарь.

Автозавершение слов Интересная функция **OpenOffice Writer** — автозавершение слов. Если Вы набрали несколько первых букв слова, которое ранее уже набирали, **Writer** автоматически подставляет это слово.

Если предложенный вариант не подходит, просто продолжайте набирать. А если он подходит, нажмите Enter — слово будет набрано полностью, и Вы сможете продолжать набор со следующего слова. Иногда данная возмож-

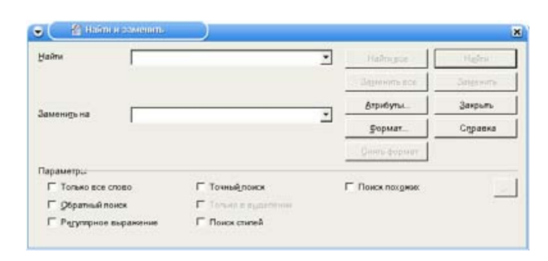


Рис. 11.7: Окно поиска и замены

ность может заметно сэкономить время, а при обычном наборе поведение **Writer** не изменяется. Если Вас раздражает автозавершение, его можно отключить. Если же система часто предлагает какое–то конкретное не устраивающее Вас слово, можно удалить именно это слово. Чтобы произвести эти операции, выберите в главном меню пункт *«Сервис»*, затем *«Автозамена/Автоформат. . . »* и щёлкните мышью на пункт *«Завершение слова»* в верхней части окна.

Чтобы отключить автодополнение, следует щелчком мыши убрать «галочку» в пункте *«Дополнять слова»*. Чтобы система не предлагала то или иное конкретное слово, его следует найти в списке в правой части окна (он отсортирован по алфавиту), выбрать щелчком мыши и удалить, нажав кнопку **«Удалить запись»**.

Поиск и замена Как и любой современный редактор текста, **OpenOffice Writer** позволяет производить поиск по тексту и замену найденной последовательности знаков на другую. Чтобы произвести поиск или замену, следует в главном меню выбрать пункт *«Правка»*, затем *«Найти и заменить. . . »*, или же нажать клавиши `Ctrl`+`F`. Появится окно поиска и замены (рис. 11.7).

В нём Вы можете ввести строку символов, которую нужно найти, и при необходимости — другую, на которую её следует заменить. Кнопка **«Найти»** позволяет найти данную строку. Кнопка **«Заменить»** заменяет найденную строку на новую и находит, где эта строка появляется в тексте в следующий раз (новое нажатие кнопки **«Заменить»** произведёт замену и поиск следующей строки и т.п.). Чтобы заменить одну строку на другую во всём тексте, используйте кнопку **«Заменить все»**.

Базовое форматирование текста Как и в большинстве известных текстовых процессоров, форматирование текста в **Writer** производится отдельно по знакам и по абзацам. К формату знака относятся шрифт, размер, подчёркивание/курсив и т.д.; к формату абзаца — отступы (горизонтальные и вертикальные), выравнивание и т.д.

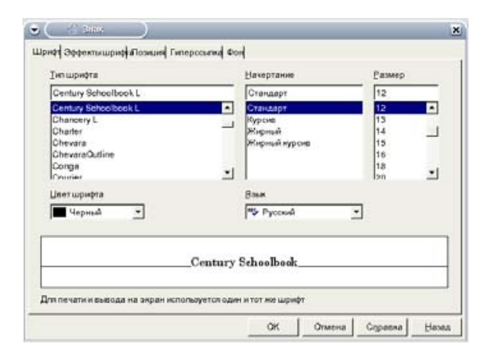


Рис. 11.8: Форматирование знаков

Форматирование знаков Для изменения формата знаков (букв) следует выделить эти знаки. Если Вы укажете изменение формата знака без выделения знаков, оно будет относиться к знакам, которые Вы введёте непосредственно после этого (не перемещая курсора).

Простейшее изменение форматирования — включение/выключение жирного текста, курсива и подчёркивания — производится щелчком по соответствующим кнопкам на панели инструментов (Ж, К, Ч). Чтобы изменить шрифт, можно выбрать его в списке, присутствующем на панели инструментов.

Для более сложного форматирования знаков следует выбрать в главном меню пункт *«Формат»*, затем *«Знаки»*. Появится окно **«Знак»** (рис. 11.8).

В этом окне, переключая пункты в верхней части окна, можно настроить все возможные варианты форматирования, относязиеся к знакам. Важно, что в данном окне также устанавливается язык текста. В **OpenOffice Writer** язык является свойством знака. Проверка орфографии каждого слова производится в соответствии с языком, указанным для него.

Форматирование абзаца Для изменения формата абзаца достаточно установить курсор в этот абзац. Чтобы изменить форматирование нескольких абзацев сразу, следует выделить их. Изменение выравнивания производится щелчком по соответствующим кнопкам на панели инструментов.

Для более сложного форматирования абзаца следует выбрать в главном меню пункт *«Формат»*, затем *«Абзац»*. Появится окно **«Абзац»** (рис. 11.9).

В этом окне, переключая пункты в верхней части окна, можно настроить все возможные варианты форматирования, относязиеся к абзацу. В частности, можно указать оформление абзацев в виде нумерованного или ненумерованного списка (пункт *«Нумерация»*).

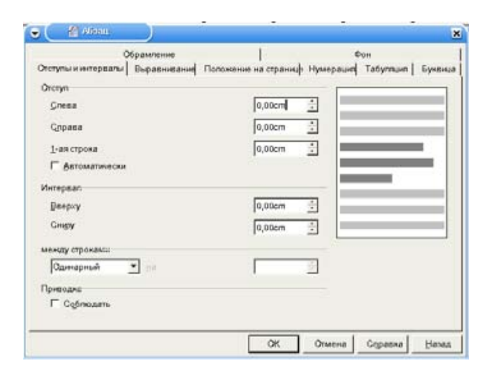


Рис. 11.9: Форматирование абзаца

Форматирование текста при помощи стилей Как и многие распространённые офисные текстовые процессоры, **OpenOffice Writer** позволяет форматировать текст при помощи стилей. Основной вид стилей — стили абзаца.

Стиль абзаца определяет форматирование как самого абзаца (отступы и т.д.), так и форматирование знаков в абзаце (шрифт, размер и т.д.).

Если абзац отформатирован при помощи стиля, изменение стиля автоматически приводит к изменению форматирования абзаца. В этом заключается первое преимущество использования стилей. Например, если требуется изменить шрифт и размер всех заголовков в тексте, без использования стилей пришлось бы вносить изменения в каждый заголовок отдельно. Но если все заголовки отформатированы при помощи стилей *«Заголовок 1»*, *«Заголовок 2»* и т.д., достаточно изменить только эти стили.

Кроме того, стили абзаца позволяют производить логическую разметку текста, т.е. указывать границы разделов, глав и т.д., а также специальные виды абзацев (например, цитаты или примеры). Начало разделов и глав определяется при помощи заголовков, выделенных стилями *«Заголовок 1»*, *«Заголовок 2»* и т.д. **Writer** может автоматически сгенерировать оглавление текста, указывая абзацы с данными стилями в качестве названий разделов. Имеется также возможность автоматической нумерации разделов.

Кроме стилей абзаца, поддерживаются стили знаков. Стиль знаков определяет форматирование знаков, не затрагивая форматирование абзацев. Стили знаков удобно использвоать, например, для выделения цитат в тексте.

Для работы со стилями используется окно **«Стилист»** (рис. 11.10).

Чтобы вызвать это окно, следует в главном меню выбрать пункт *«Формат»*, затем *«Стилист»*, или же нажать клавишу `F11`. Первоначально в окне **«Стилист»** показывается список стилей абзаца.

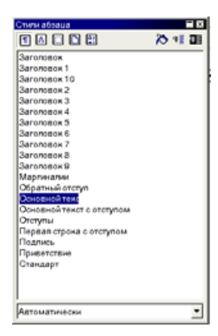


Рис. 11.10: Окно работы со стилями

Применение стилей Чтобы указать стиль форматирования абзаца, следует установить курсор на этот абзац, а затем выбрать нужный стиль щелчком мыши в окне **«Стилист»**. (Если курсор установлен на пустом абзаце, стиль всё равно будет присвоен, и набираемый далее текст будет отформатирован при помощи этого стиля).

Чтобы указать стиль для нескольких абзацев сразу, нужно выделить эти абзацы, а затем выбрать нужный стиль щелчком мыши в окне **«Стилист»**.

Для применения стилей знаков следует переключить окно **«Стилист»** в режим списка стилей знаков. Для этого щёлкните левой кнопкой мыши по иконке с буквой [A] в этом окне. После этого, чтобы отформатировать участок текста при помощи стиля знаков, следует выбрать этот участок, а затем щелчком мыши в окне **«Стилист»** указать нужный стиль. Если Вы укажете стиль знаков без выделения, оно будет относиться к знакам, которые Вы введёте непосредственно после этого (не перемещая курсора). Чтобы переключить окно **«Стилист»** обратно в режим списка стилей абзаца, следует выбрать в этом окне иконку с буквой, схожей с [П].

Редактирование стилей Для изменения любого из стилей (абзаца или знаков) следует выбрать этот стиль в окне **«Стилист»** щелчком мыши, а затем нажать правую кнопку мыши и в появившемся меню выбрать пункт *«Изменить»*. Появится окно редактирования стиля (рис. 11.11).

Пункты в верхней части окна позволяют изменить различные свойства форматирования знаков и (для стиля абзаца) свойства форматирования абзаца. Пункт *«Организация»* используется для установки свойств стиля. Поле *«Имя»* определяет название стиля. Поле *«Связан с»* указывает базовый стиль для данного стиля; если изменяется базовый стиль, аналогичные изменения автоматически вносятся и в данный стиль.

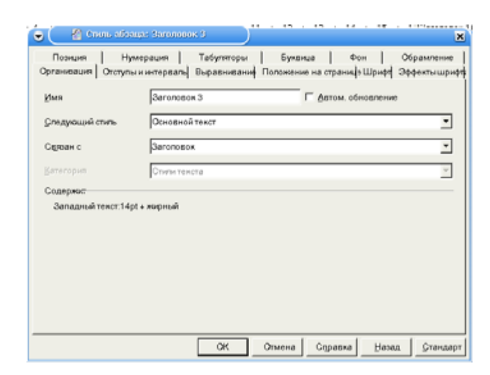


Рис. 11.11: Редактирование стилей

Для стилей абзацев имеется также поле *«Следующий стиль»*. В нём можно указать, каким стилем автоматически форматируется следующий абзац (при вводе текста), если предыдущий абзац отформатирован данным стилем. Так, для стиля заголовка обычно устанавливается следующий стиль *«Основной текст»*; это позволяет после ввода заголовка сразу, не тратя время на выбор стиля, вводить текст. Чтобы создать новый стиль, следует щёлкнуть правой кнопкой мыши в окне **«Стилист»** и выбрать в появившемся меню пункт *«Создать»*. Появится окно редактирования стиля, в котором Вы сможете указать все необходимые свойства стиля. Если окно **«Стилист»** находится в режиме списка стилей абзаца, то создаётся стиль абзаца; если же оно находится в режиме списка стилей знаков, то создаётся стиль знаков.

Кроме того, система позволяет отформатировать абзац обычными средствами (см. с. 184), а затем автоматически создать стиль с именно таким форматированием. Для этого, отформатировав абзац, нажмите в окне **«Стилист»** вторую справа иконку в правом верхнем углу. Система предложит ввести название нового стиля, после чего он будет создан и появится в списке. Вы можете также удалить созданный ранее стиль. Для этого следует выбрать стиль в окне **«Стилист»** щелчком мыши, а затем нажать правую кнопку мыши и в появившемся меню выбрать пункт *«Удалить»*. Однако система не позволяет удалить стили, которые присутствуют в **Writer** изначально.

Форматирование страницы **Writer** позволяет настраивать размер страницы, а также верхний и нижний колонтитулы (т.е. строки вверху и внизу страницы) и другие свойства форматирования страницы. Более того, для разных страниц можно установить разное форматирование при помощи стилей страниц.

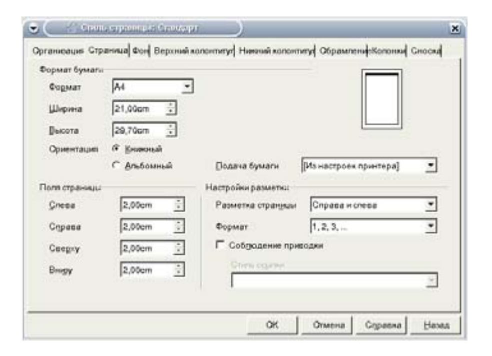


Рис. 11.12: Редактирование стиля страницы

Установка формата страницы Чтобы установить формат страницы, включить верхний или нижний колонтитул и т.п., вызовите в главном меню пункт *«Формат»*, затем *«Страница»*. Появится окно редактирования стиля страницы (рис. 11.12).

В данном окне можно установить, в частности:

- Размер страницы (пункт *«Страница»* в верхней части окна);
- Стиль нумерации страницы — арабские цифры, римские цифры и т.д. Пункт *«Страница»*, поле *«Настройки разметки»* — *«Формат»*;
- Включение и отступ верхнего и нижнего колонтитулов — пункты *«Верхний колонтитул»* и *«Нижний колонтитул»*. В частности, в любом из этих пунктов можно отключить «галочку» *«Одинаковое содержимое слева/справа»*; это позволит создать разные колонтитулы для чётных и нечётных страниц.;
- Включение и вид рамки вокруг страницы — пункт *«Обрамление»*;
- Форматирование текста на странице в несколько колонок — пункт *«Колонки»*.

Установив необходимый формат страницы, нажмите кнопку **«OK»**.

Нумерация страниц Чтобы включить нумерацию страниц, следует прежде всего включить необходимый колонтитул (верхний либо нижний). Включив колонтитул, Вы сможете редактировать его (при режиме просмотра текста *«Разметка страницы»*) как обычный текст.

Установив курсор в колонтитул, выберите в главном меню пункт *«Вставка»*, затем *«Поле»*, затем *«Номер страницы»*. Появится номер страницы. На каждой странице он будет автоматически появляться и устанавливаться в нужное значение.

Чтобы передвинуть номер страницы (вместе с остальным текстом колонтитула) в левый угол, центр или правый угол, следует, установив курсор в колонтитул, переключить выравнивание абзаца (по левому краю, центру или правому краю) соответствующей кнопкой на панели инструментов **Writer**.

Если номер должен быть по-разному расположен на чётной и нечётной странице, следует в окне редактирования стиля страницы (см. с. 187) отключить для колонтитула, в котором находится номер страницы, «галочку» *«Одинаковое содержимое слева/справа»*. После этого расположите номер нужным образом в колонтитуле на одной чётной и одной нечётной странице.

Стили страниц Во многих случаях требуется обеспечить разное форматирование разных страниц — например, отключить номер страницы для титульного листа.

OpenOffice Writer позволяет устанавливать полностью независимое форматирование для разных страниц — вплоть до различного размера листа. Для этого используются стили страницы.[2]

Работа с главами текста Если заголовки глав отмечены стилями ряда *«Заголовок»*, причём уровень заголовков отражён в использовании стилей (т.е., например, для разделов верхнего уровня используется *«Заголовок 1»*, для подразделов — *«Заголовок 2»* и т.д.), **OpenOffice Writer** позволяет автоматически производить нумерацию глав, а также создавать оглавление.

Нумерация глав Чтобы произвести автоматическую нумерацию глав, выберите в главном меню пункт *«Сервис»*, затем *«Нумерация глав»*. Появится окно **«нумерация глав»** (рис. 11.13).

Здесь Вы можете настроить нумерацию глав. В правой части окна показывается пример нумерации в том виде, который настроен в данный момент. Можно найти желаемый вариант нумерации, пробуя различные настройки и наблюдая за изменениями примера.

Генерация оглавления Для автоматического создания оглавления установите курсор в точку текста, где должно быть оглавление. Выберите в главном меню

[2]Использование стилей страниц — достаточно сложная операция, выходящая за пределы данного краткого руководства.

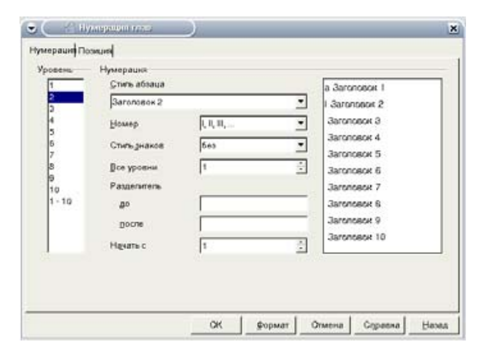


Рис. 11.13: Настройка нумерации глав

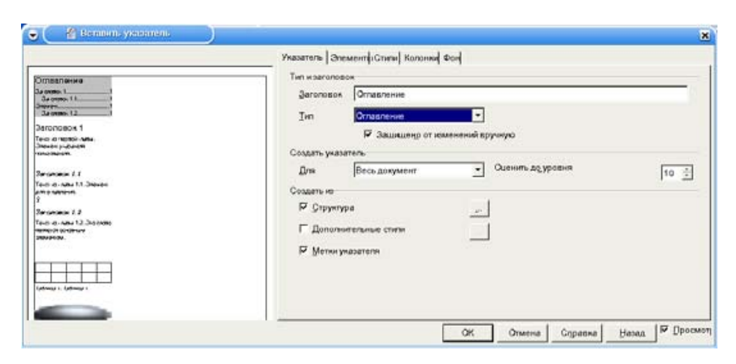


Рис. 11.14: Создание оглавления

пункт *«Вставка»*, затем *«Указатели»*, затем *«Указатели. . . »*. Появится окно **«Вставить указатель»** (рис. 11.14)

В данном окне можно осуществить вставку различных указателей, но первоначально предлагаемый вариант — именно оглавление. В поле *«Оценить до уровня»* Вы можете указать, заголовки каких уровней следует включать в оглавление.

Например, при значении 3 в оглавление будут включены все абзацы со стилем *«Заголовок 1»*, *«Заголовок 2»* и *«Заголовок 3»*. После нажатия кнопки **«OK»** система сгенерирует оглавление и поместит его там, где располагался курсор.

Вставка рисунков **OpenOffice Writer** позволяет вставлять в текст рисунки (взятые из файлов) и указывать их расположение, а также то, будут ли они «об-

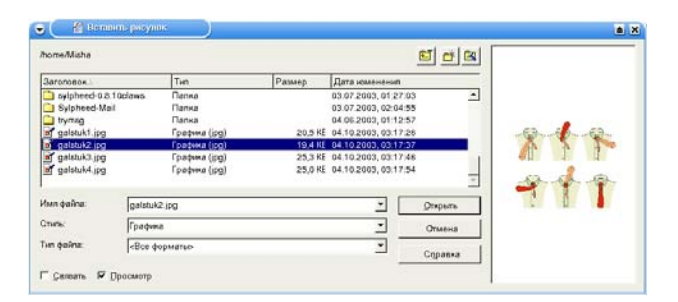


Рис. 11.15: Вставка рисунка из файла

текаться» текстом. Чтобы вставить рисунок в текст, выберите в главном меню пункт *«Вставка»*, затем *«Рисунок»*, затем *«Из файла...»* Появится окно вставки рисунка из файла (рис. 11.15).

Здесь Вы можете выбрать нужный файл.

При выборе файла щелчком мыши (если включена «галочка»*«Просмотр»*) включается его предварительный просмотр в правой части окна. Чтобы вставить выбранный рисунок в текст, нажмите кнопку **«Открыть»**.

Щёлкнув по рисунку правой кнопкой мыши, Вы можете вызвать меню настройки рисунка. С помощью этого меню можно установить, в частности:

- Как будет располагаться рисунок в тексте (пункт *«Привязка»*) — на фиксированной позиции по отношению к странице, к определённому абзацу, к определённому знаку или же как знак. Если рисунок располагается «как знак», он вставляется в строку так же, как и обычный знак (но его размер не меняется при изменении размера шрифта).;
- Будет ли текст «обтекать» рисунок и если да — каким именно образом (пункт *«Обтекание»*).

Сохранение и чтение файлов **OpenOffice Writer** сохраняет и считывает файлы в формате `.sxw`. Этот формат полностью опубликован и основан на открытых стандартах. Кроме того, он позволяет сохранять и считывать файлы в форматах `.rtf` (Rich Text Format), `.doc` (Microsoft Word 95 или 97/2000/XP) и других.

Сохранение файла Чтобы сохранить текущий текст на диске в виде файла, следует выбрать в главном меню пункт *«Файл»*, затем *«Сохранить»*, либо нажать клавиши `Ctrl`+`S`. Если файл уже был сохранён, он будет вновь со-

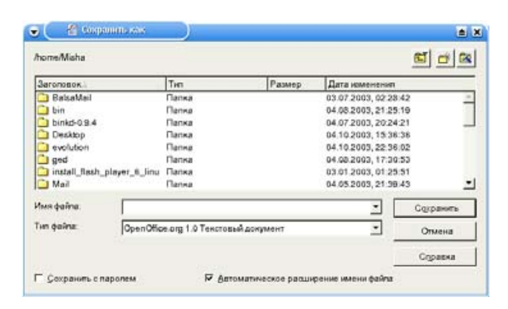


Рис. 11.16: Сохранение файла

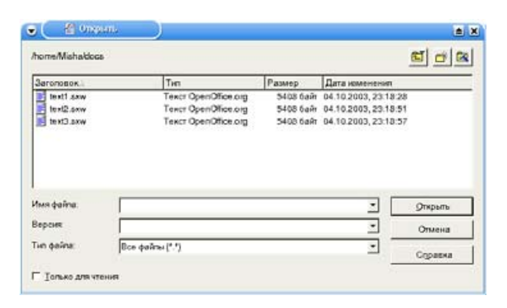


Рис. 11.17: Чтение файла

хранён под тем же именем. Если же он ещё не был сохранён, появится окно выбора имени и формата сохраняемого файла (рис. 11.16).

В данном окне можно выбрать нужный каталог для сохранения файла, указать имя файла, а также выбрать его формат (тип). Указав имя и тип файла, следует нажать кнопку **«Сохранить»**. После этого файл будет записан на диск.

Сохранение файла под новым именем Если Вы работаете с файлом, который ранее уже был сохранён, и хотите сохранить его под другим именем (или же в другом формате — например, `.rtf` или `.doc`), выберите в главном меню пункт *«Файл»*, затем *«Сохранить как. . . »* Появится окно выбора имени файла (см. рис. 11.16), и Вы сможете указать новые имя и формат файла.

Чтение файла Чтобы открыть (считать) файл с диска, следует выбрать в главном меню пункт *«Файл»*, затем *«Открыть»*, либо нажать клавиши `Ctrl`+`O`. Появится окно выбора файла (рис. 11.17).

В данном окне можно найти нужный каталог и выбрать в нём файл, который требуется открыть. Указав щелчком мыши нужный файл, нажмите кнопку **«Открыть»**. **Writer** считает файл и, если чтение прошло успешно, Вы сможете просматривать и редактировать содержимое данного файла.

11.2 Редактирование HTML

Редактирование файлов HTML (формата для WWW-страниц) в **OpenOffice** весьма сходно с редактированием текстов в **OpenOffice Writer**. (Реально редактирование HTML производится именно **Writer**, работающим в особом режиме). Это позволяет создавать WWW-страницы без специального обучения, а также легко преобразовывать текстовые документы в WWW-страницы. Вы можете использовать для редактирования HTML-документов практически все приёмы, описанные выше для **Writer**.

Базовое редактирование HTML Документы HTML можно форматировать так же, как и обычные документы. Однако при этом получившиеся WWW-страницы будут достаточно неудобны для просмотра. Для корректного форматирования документов HTML следует использовать специальные стили абзаца, которые автоматически предлагаются в окне **«Стилист»** при редактировании такого документа.

Для основного текста применяйте стиль *«Основной текст»*, для заголовков — стили группы *«Заголовок»*, для цитат — *«Цитата»* и т.д. Можно указывать выравнивание абзацев, а также выделять участки текста жирным шрифтом, курсивом или подчёркиванием. А вот изменять шрифт очень нежелательно — это может привести к сложностям у некоторых пользователей при просмотре Вашей страницы.

Создание ссылок Как известно, важный элемент HTML — возможность создания ссылок (hyperlinks) на другие документы.

Чтобы создать ссылку, следует установить курсор на точку в тексте, где она должна быть, и выбрать в главном меню пункт *«Вставка»*, затем *«Гиперссылка»*. Появится окно создания ссылки (рис. 11.18).

В данном окне следует указать документ, на который производится ссылка, а также текст ссылки. Текст ссылки указывается в поле *«Текст»*. Способ указания документа, на который производится ссылка, переключается в левой части окна.

При выборе пункта *«Интернет»* предоставляется возможность ввести ссылку на документ в Интернете в полном формате (URL). Пункт *«Письма и сообщения»* позволяет описать ссылку на адрес электронной почты (`mailto`) или

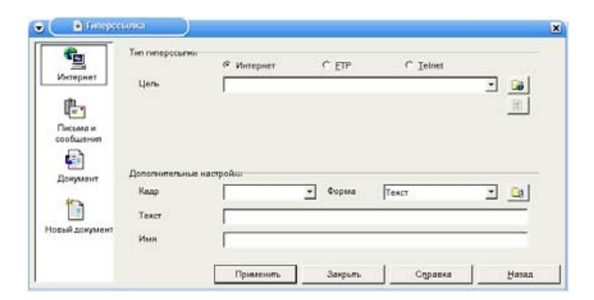


Рис. 11.18: Создание ссылок

ньюс-группу **usenet** (`news`). Выбрав пункт *«Документ»*, можно вставить ссылку на существующий документ на диске. Следует учитывать, что хотя в поле *«Путь»* указывается полный путь к документу, реально, если документы находятся в одном каталоге, создаётся ссылка только на имя документа — поэтому в случае переноса обоих файлов в Интернет ссылка продолжает работать.

Наконец, пункт *«Новый документ»* позволяет создать ссылку на документ, который ещё не создан на диске. Система предоставляет возможность немедленно создать этот документ (для этого следует выбрать пункт *«Править сейчас»* в верхней части окна) или же создать ссылку, не создавая документа (пункт *«Править позже»*). Важно корректно выбрать тип файла из списка (например, *«Документ HTML»*), а не только указать полное имя файла. Когда данные для ссылки введены, нажмите кнопку **«Применить»**. Ссылка будет создана.

Чтобы отредактировать ссылку, следует установить на неё курсор и выбрать в главном меню пункт *«Вставка»*, затем *«Гиперссылка»*. Появится такое же окно (см. рис. 11.18), но заполненное, с данными указанной ссылки. В этом окне можно произвести нужные изменения и нажать кнопку **«Применить»**.

Сохранение и чтение файлов HTML Сохранение и чтение при редактировании файлов HTML производится точно так же, как и в **Writer** (см. с. 191). Важно, что в составе файлов HTML рисунки не сохраняются. Поэтому при переносе файла на другой компьютер или в Интернет они могут быть потеряны. Даже если на другой компьютер перенесён и файл, и рисунки к нему, но они находятся в разных каталогах, рисунки могут исчезнуть из документа. Чтобы этого не произошло, лучше всего держать все рисунки, вставляемые в документ, в том же каталоге, что и сам документ, и переносить их на другой компьютер или в Интернет вместе с документом. Следует учитывать, что **OpenOffice** может считать практически любой файл в формате HTML, но не всегда при этом будут корректно показаны русские буквы. Это не ошибка в **OpenOffice**, а недостаток таких файлов — в них не указана или некоррект-

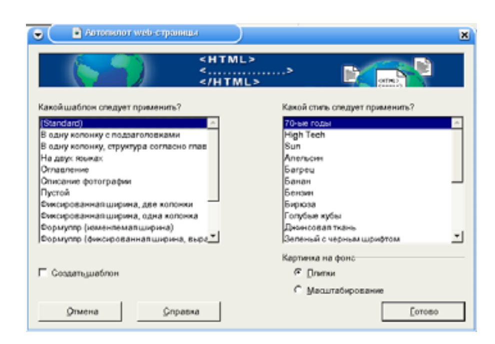


Рис. 11.19: Автопилот web-страницы

но указана кодировка русских букв. Чтобы указать вид кодировки, следует переключиться в режим редактирования исходного текста HTML (см. ниже) и ввести или отредактировать в начале файла (между условными строками <HEAD> и </HEAD>) строку следующего вида:

```
<META HTTP-EQUIV="CONTENT-TYPE" CONTENT="text/html; charset=koi8-r">
```

Вместо `koi8-r` должна быть указана кодировка русских букв в данном файле; наиболее распространённые кодировки — `koi8-r`, `windows-1251` и `utf8`.

Редактирование исходного текста HTML **OpenOffice** позволяет переключиться с редактирования содержимого HTML-документа в визуальном режиме (т.е. в том виде, в котором оно будет видимо в веб-браузере) на работу с исходным текстом на языке HTML. Это может быть полезно в целом ряде случаев, когда требуется проверка и исправление непосредственно HTML-текста. Чтобы включить или отключить режим редактирования исходного текста HTML, выберите в главном меню пункт *«Вид»*, затем *«Исходный текст HTML»*.

11.2.1 Создание WWW–страниц при помощи Автопилота

OpenOffice содержит *«Автопилот WWW–страниц»*, позволяющий в короткий срок и без дополнительных затрат усилий создать стандартную и достаточно красивую WWW-страницу. Чтобы воспользоваться *«Автопилотом»*, выберите в главном меню пункт *«Файл»*, затем *«Автопилот»*, затем *«Web-страница. . . »* Появится окно **«Автопилот web-страницы»** (рис. 11.19).

Выбирая значения полей, Вы сможете сразу же увидеть на экране, как будет выглядеть созданная страница. Подобрав желаемый вид, нажмите кнопку **«Готово»**. Вы сможете редактировать HTML-документ, сразу обладающий

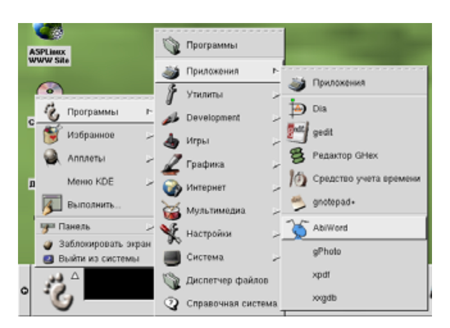


Рис. 11.20: Запуск AbiWord из стартового меню *GNOME*

нужным внешним видом. Теперь достаточно лишь наполнить его информацией.

11.3 Текстовый процессор AbiWord

Программа **AbiWord** представляет собой именно типичный процессор слов, которая постоянно находится в процессе развития (в дистрибутив включена версия 2.0) и потому еще не вполне стабильна. Однако и в нынешнем своем виде вполне пригодна для использования, в том числе и в русскоязычной среде.

AbiWord запускается из командной строки терминала (или из минитерминала) одноименной командой `abiword`. Кроме того, он может быть вызван из стартового меню *GNOME* (*«Программы»- «Приложения»- «AbiWord»*, рис. 11.20) и *KDE* (*«Приложения»- «AbiWord»*, рис. 11.21).

При запуске **AbiWord** открывается окно с пустым безымянным файлом в рабочей области (рис. 11.22). В верхней части окна — строка главного меню и три инструментальные панели — Стандартная, Панель форматирования и Дополнительная. В нижней части — строка состояния. На рабочем поле выведены линейки, горизонтальная и вертикальная. Это вид **AbiWord** по умолчанию, он может быть изменен соответствующими настройками.

Главное меню **AbiWord** включает пункты *«Файл»*, *«Редактирование»*, *«Вид»*, *«Вставка»*, *«Форматирование»*, *«Сервис»*, *«Окно»*, *«Помощь»*.

В меню *«Файл»* доступны следующие действия: создание файла, открытие существующего, сохранение, сохранение под другим именем, установка параметров страницы, печать и выход (рис. 11.23).

Наибольший интерес представляют пункты *«Открыть»* и *«Сохранить как...»*, поскольку именно ими определяется потенциал **AbiWord** по обмену докумен-

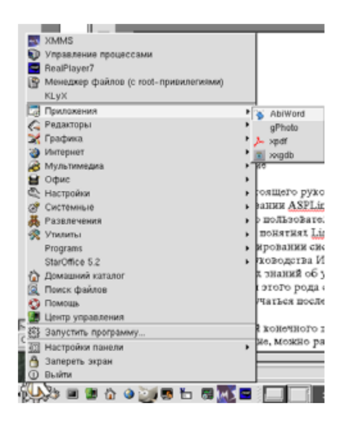


Рис. 11.21: Запуск AbiWord из стартового меню *KDE*

тами с другими приложениями.

AbiWord способен считывать документы в следующих форматах (рис. 11.24): собственном (`*.abw`), в том числе и автоматически сжатом (`*.zabw`), DocBook (`*.dbk`), Microsoft Word (`*.doc`), XHTML (`*.html`), Rich Text Format (`*.rtf`), UTF8 (`*.utf8`), WML (`*.wml`) и текстовом (`*.txt`).

Сохранение созданных `AbiWord` документов, кроме собственных форматов `*.abw` и `*.zabw`, возможно в виде файлов DocBook, HTML, RTF, UTF8, WML (`*.wml`), текстовом, а также в формате LaTeX (рис. 11.25). Кроме того, `AbiWord` имеет возможность взаимного обмена документами с такими КПК, как Palm и Psion.

AbiWord корректно работает с кириллическими документами в текстовом формате.

Возможно также считывание документов MS Word, RTF и HTML с символами кириллицы, однако в этих случаях возможны как потеря форматирования (или его искажение), так и отказы считывания. Однако в целом работа с кириллицей улучшается с каждой новой версией **AbiWord**.

Меню *«Редактирование»* (рис. 11.26) объединяет действия по выделению, вырезанию, копированию и вставке текстовых фрагментов, поиску, замене и переходу, отмене и возврату выполненных операций.

Через меню *«Вид»* осуществляется включение или отключение элементов интерфейса (рис. 11.27) — инструментальных панелей, линеек на рабочем полу, строки статуса.

Здесь же устанавливается показ непечатаемых спецсимволов, колонтитулов, устанавливается масштаб показа документа (75, 100, 200%, по ширине или

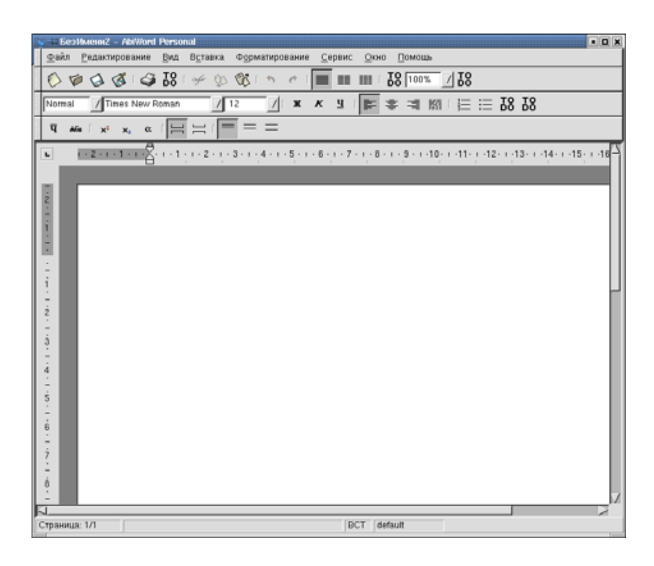


Рис. 11.22: AbiWord, вид по умолчанию.

по размеру страницы, а также произвольный с шагом в 1%, рис. 11.28) и происходит переключение в полноэкранный режим, в котором отключены все элементы интерфейса, кроме строки главного меню (рис. 11.29).

Через меню *«Вставка»* (рис. 11.30) выполняются:

- вставка разрыва страницы или раздела,
- нумерация страниц,
- вставляется дата и (или) время в различных форматах, а также поля даты, приложения, номера,
- вставка символов из кодовой таблицы с учетом шрифта (аналог `Character Map` в Windows),
- вставка рисунков в форматах PNG, BMP, SVG (Scalable Vector Graphics).

В меню *«Форматирование»* (рис. 11.31) определяются:

- гарнитура, начертание и кегль шрифта,
- параметры абзаца (отступы, интервалы и выравнивание),
- формат списков (нумерованных и маркированных) и вид маркеров для последних,

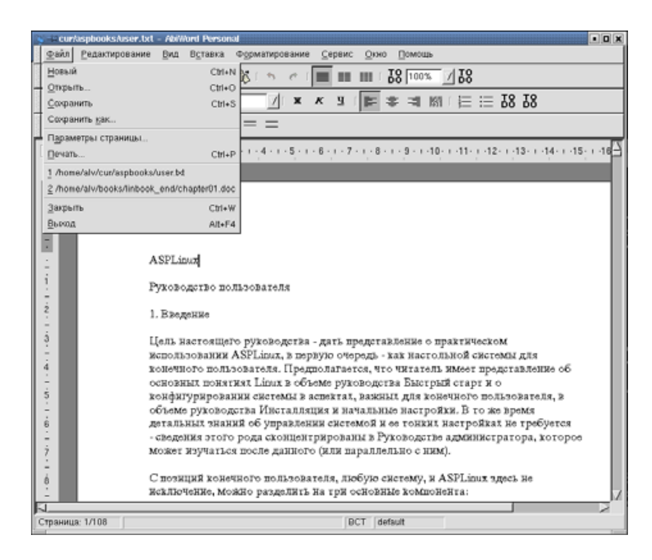


Рис. 11.23: Меню *«Файл»*

- параметры страницы (формат бумаги, ориентация, поля),
- многоколоночное (одна, две или три колонки) представление текста и величина табуляции.

Предполагается возможность стилевого оформление текстов, но эта функция пока не реализована.

В меню *«Сервис»* три пункта (рис. 11.32):

- Правописание, где можно вызвать программу проверки орфографии или включить автоматическую проверку;
- Статистика, где осуществляется подсчет слов, абзацев, символов и т.д., с опцией автообновления через фиксированный промежуток времени;
- Настройки, где включается/выключается показ инструментальных панелей, линеек и т.д.

В меню *«Окно»* можно создать новое (пустое) окно или просмотреть список открытых документов — каждый из них выводится в собственном окне со всеми управляющими элементами. В меню *«Помощь»* вызывается система помощи **AbiWord**.

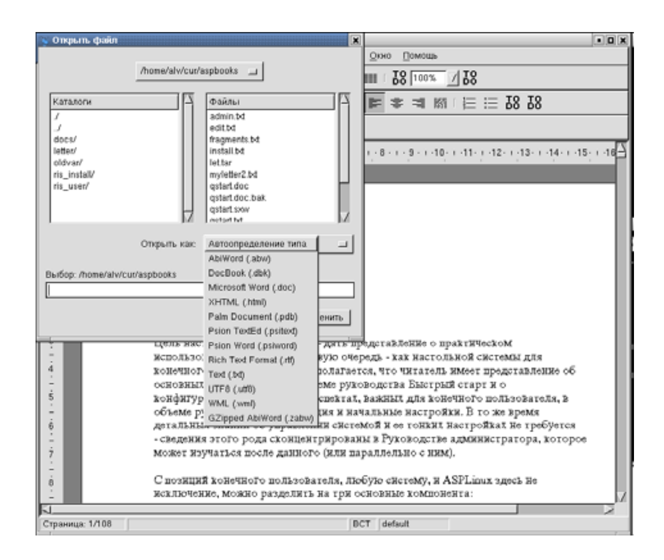


Рис. 11.24: Форматы документов, доступные AbiWord для чтения

Многие доступные через меню действия выполняются также с помощью кнопок инструментальных панелей. В стандартной панели собраны кнопки, отвечающие за создание, открытие и сохранение (в том числе и под другим именем) файла, печать, проверку орфографии, вырезание, копирование и вставку текстовых фрагментов, отмену и возврат операций, включение многоколоночного представления и масштабирование документа.

Выпадающими списками Панели форматирования устанавливаются структурные элементы документа (заголовки, блочный текст, нормальный текст), гарнитура и кегль шрифта, кнопками — его начертание, выравнивание абзаца, включаются нумерованные и маркированные списки.

Кнопками Дополнительной панели устанавливаются шрифтовые эффекты (подчеркивание и перечеркивание), верхние и нижние индексы, горизонтальные отступы перед абзацами и междустрочные интервалы.

По щелчку правой кнопкой мыши на рабочем поле доступно контекстное меню, через которое можно вырезать, скопировать или вставить текстовый фрагмент, а также установить шрифтовое оформление и параметры абзаца (рис. 11.33).

В целом **AbiWord** не может заменить полнофункциональный текстовый процессор, но вполне пригоден для составления деловых документов небольшого объема и несложного оформления.

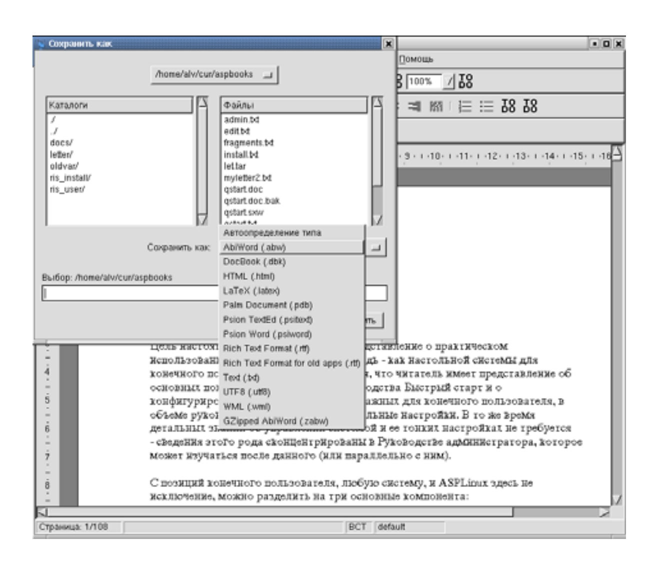


Рис. 11.25: Форматы документов, доступные AbiWord для сохранения

11.4 Интегрированный пакет KOffice

В состав **ASPLinux** входит еще один интегрированный пакет — **KOffice**, один из компонентов среды *KDE*.

Интегрированный пакет **KOffice** включает в себя следующие приложения:

- текстовый процессор **KWord**,
- электронную таблицу **KSpread**,
- векторный графический редактор **Kontour**,
- программу подготовки презентаций **KPresent**,
- программу для построения диаграмм **KChart**,
- программу для работы с математическими формулами **KFormula**,
- настольную издательскую систему **KIvio**.

Каждая из этих программ может быть запущена самостоятельно, через стартовое меню *KDE* (пункт *«Офис»*) или из строки терминала (минитерминала) одноименной командой (в нижнем регистре): `kword` — для текстового процессора, `kspread` — для электронной таблицы и т.п.

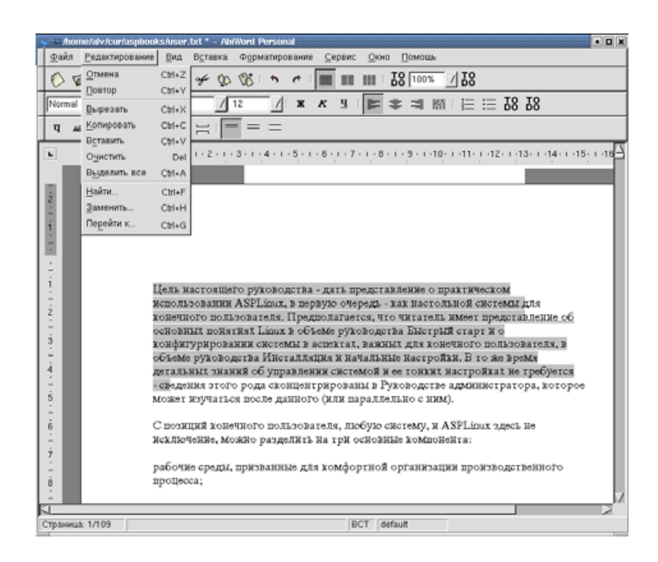


Рис. 11.26: Действия через меню *«Редактирование»*

Однако более эффективно использование **KOffice** с собственной интегрирующей оболочкой — Рабочим столом **KOffice**. Оболочка эта достаточно компактна и проста в использовании. Она образована окном с контекстно-чувствительными главным меню и инструментальными панелями (их вид зависит от текущего приложения пакета), рабочего поля и расположенных слева от него переключаемых панелей — Объектов и Документов (рис. 11.34).

Через панель объектов запускаются (в пределах одного и того же рабочего поля) все перечисленные выше приложения, входящие в комплект **KOffice**. Панель же документов служит для переключения между открытыми документами (рис. 11.35).

Пакет **KOffice** находится в стадии активного развития, не все его компоненты равноценны с точки зрения устойчивости и функциональности. И потому ниже будут рассмотрены только основные из них.

Поскольку традиционно основой офисных пакетов являются текстовые редакторы, начать рассмотрение следует с программы **KWord**. При ее автономном (через стартовое меню или командой `kword`) запуске возникает панель с предложением открыть существующий документ или создать новый (рис. 11.36).

Кроме собственного формата (`*.kwd`, основанного на стандарте XML), `KWord` способен считывать файлы HTML, MS Word 97 и текстовые. Стоит обратить внимание, что корректное преобразование кириллических докумен-

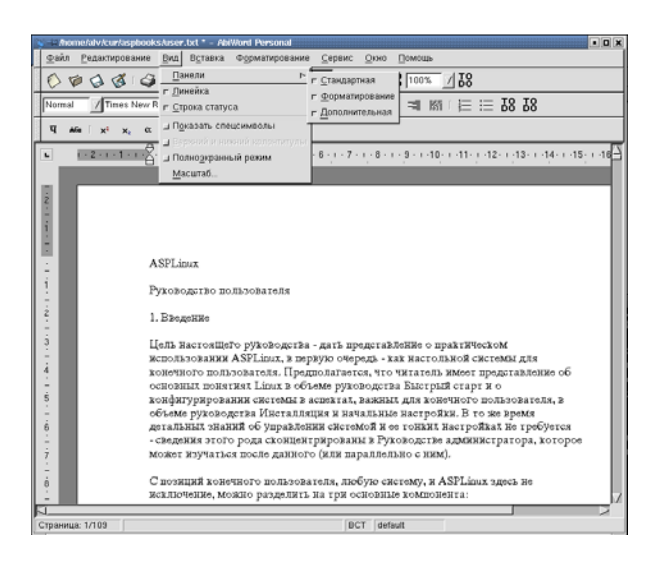


Рис. 11.27: Действия через меню *«Вид»*

тов MS Word осуществляется только для последних (MS Word 97 и выше) версий. При создании нового документа можно использовать несколько предопределенных шаблонов, учитывающих формат страницы (см. рис. 11.36).

После открытия или создания документа открывается окно **KWord**, включающее рабочее поле, строку меню и полный набор инструментальных панелей (рис. 11.37). Панели инструментов разбиты на многочисленные группы по назначению (файловые операции, редактирование, форматирование, работа с таблицами, формулами, врезками, изображениями и т.д.). Набор их таков, что практически все действия могут быть выполнены без обращения к главному меню.

Подробное рассмотрение инструментальных панелей не представляется це-

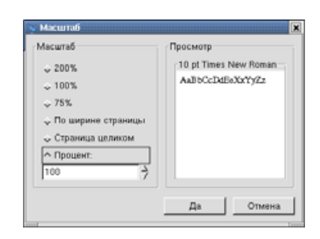


Рис. 11.28: Масштабирование показа документа

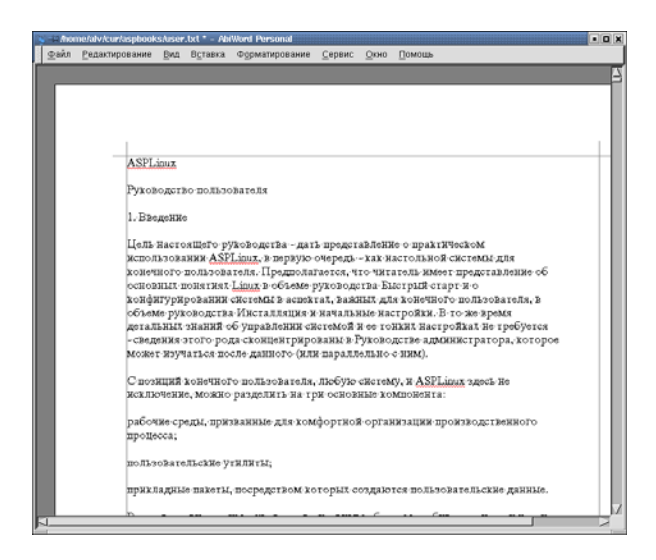


Рис. 11.29: AbiWord — полноэкранный режим

лесообразным - при их многочисленности назначение большинства элементов вполне прозрачно. Вместо этого остановимся на особенностях подготовки документов в **KWord**.

Отличительная особенность **KWord** — четкая ориентация на стилевое оформление документа. Набор предопределенных стилей построен очень логично. Так, он включает отдельные стили для заголовков различных уровней (`Head 1`, `Head 2`, `Head 3`) и для их содержания (*«Contents Head 1»* и т.д.). Предусмотрены стили для списков нумерованных, алфавитных и маркированных.

Предопределенный набор стилей не очень обширен, но легко может быть изменен и дополнен. Для этого предназначен *«Менеджер стилей»*, вызываемый через пункт главного меню *«Дополнительно»*. Он вызывает панель со списком всех доступных стилей и управляющими кнопками для манипуляции ими (рис. 11.38).

С помощью кнопки **«Добавить»** можно создать собственный стиль "с нуля", определив для него (рис. 11.39):

- гарнитуру, начертание и кегль шрифта (кнопка Шрифт),
- цвет шрифта,
- параметры абзаца (кнопки **«Отступы»**, **«Интервалы»** и **«Выравнивание»**),

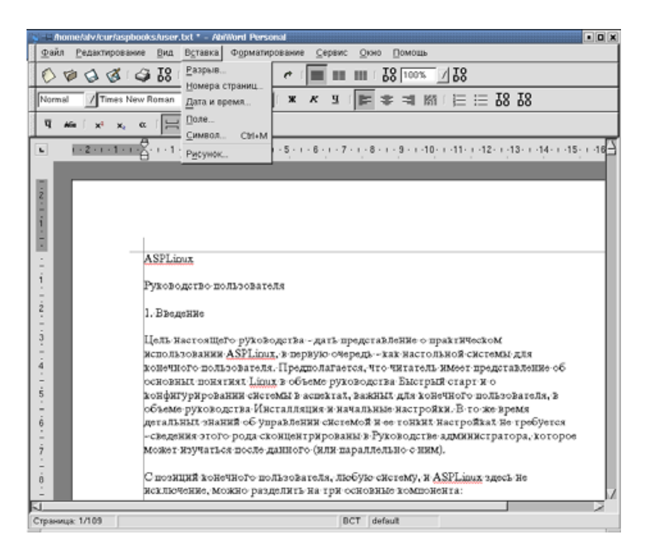


Рис. 11.30: Меню *«Вставка»*

- формат рамки, обрамляющей текстовый блок данного стиля (кнопка **«Рамка»**),
- формат нумерованных и алфавитных списков,
- параметры табуляции.

Те же параметры могут быть изменены (кнопка **«Изменить»**) для любого из предопределенных шрифтов. Кроме того, существующий стиль может быть скопирован и сохранен под другим именем с некоторыми измененными параметрами. Созданные пользователем стили могут быть уничтожены (кнопкой **«Удалить»**), но для базовых предопределенных стилей (таких, как `Standard`, **«Head #»**, списки разного вида) удаление невозможно.

Созданные в **KWord** документы могут быть сохранены, кроме собственного формата, также в виде html- и текстовых файлов. Работа с кириллическими текстами в формате **KWord** сложностей не вызывает, но корректный экспорт русскоязычных документов возможен только в текстовый формат.

Электронная таблица **KSpread** при запуске также запрашивает открытие существующего файла или создание нового. Среди считываемых форматов, кроме собственного (`*.ksp`) — таблицы Excel 97 и CSV (текстовый с разделением полей). При считывании последнего следует запрос о разделителе (точка, точка с запятой, табуляция), после чего таблица открывается (рис. 11.40).

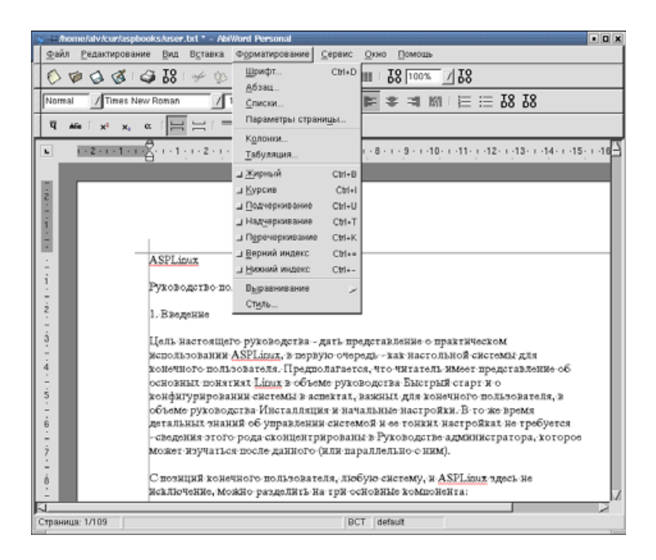


Рис. 11.31: Возможности меню Форматирование

Над ячейками таблицы возможны основные действия — изменение формата, ввод формул и математических функций и т.д. Многие опции (например, построение диаграмм) пока не реализованы, но для несложных расчетов **KSpread** применяться может. Результаты работы сохраняются либо в таблице собственного формата (допускающего многостраничные рабочие книги), либо в виде текстового файла CSV.

Кроме **KWord** и **KSpread**, из программ пакета **KOffice** заслуживает внимания векторный редактор **Kontour**, о котором будет рассказано в следующей главе.

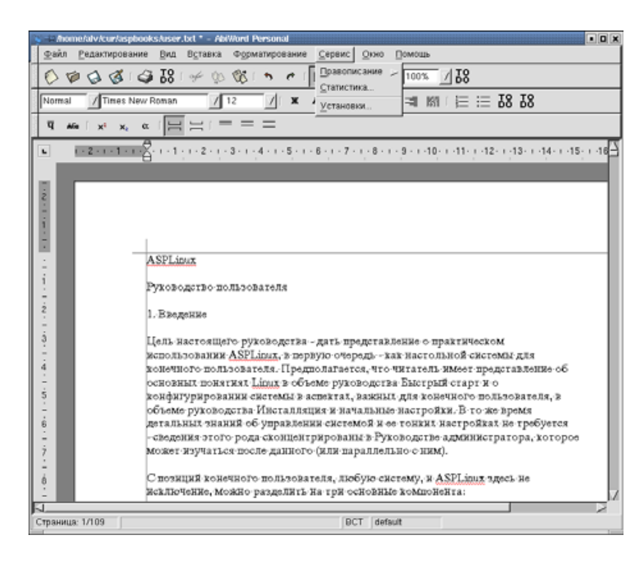


Рис. 11.32: Меню *«Сервис»*

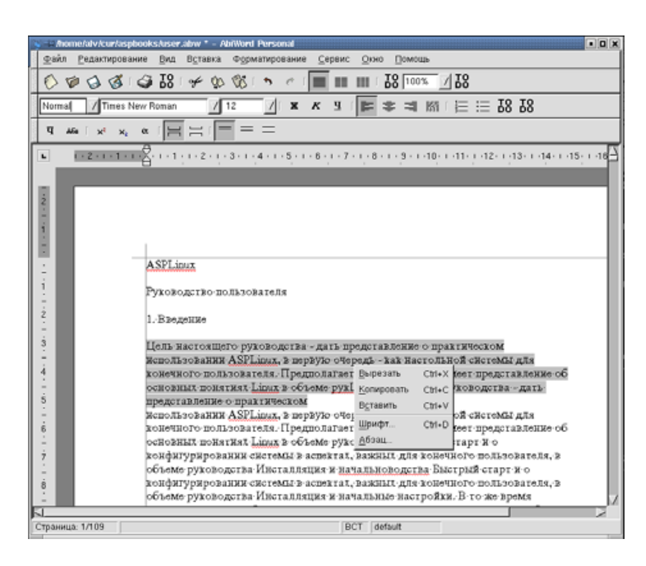


Рис. 11.33: Контекстное меню рабочего поля

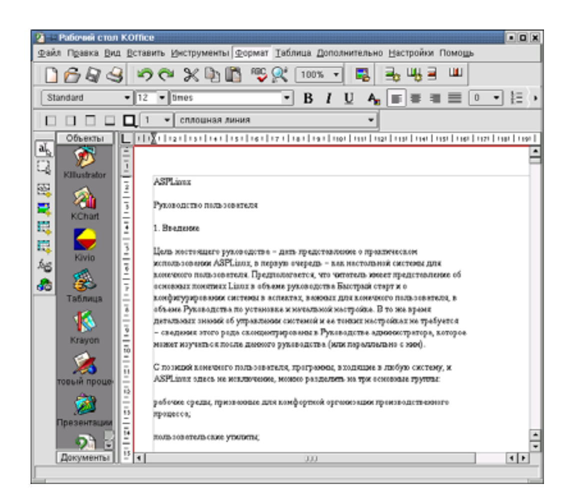


Рис. 11.34: Рабочий стол **KOffice** с открытым документом **KWord**

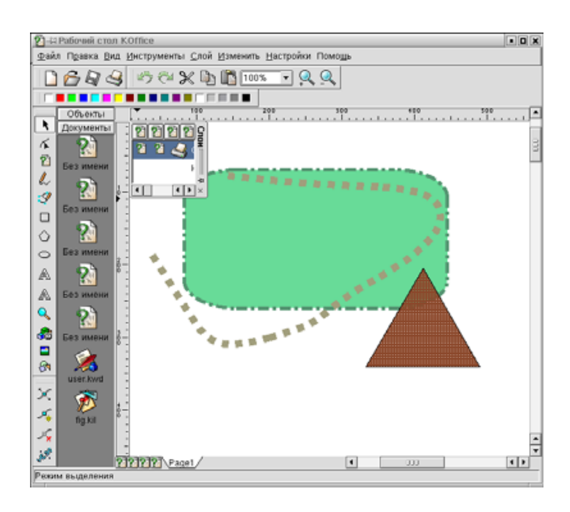


Рис. 11.35: Панель Документы рабочего стола **KOffice**

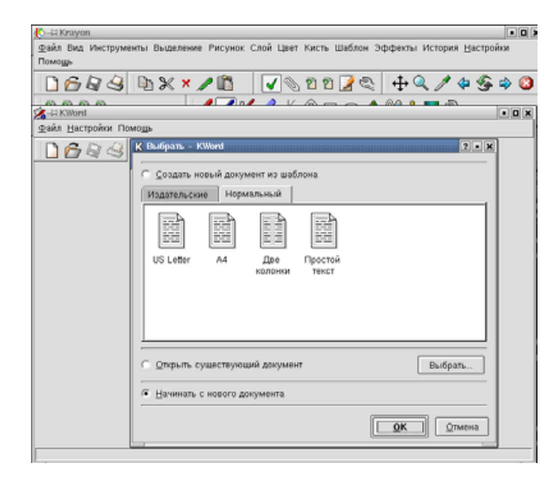


Рис. 11.36: Диалоговая панель при запуске **KWord**

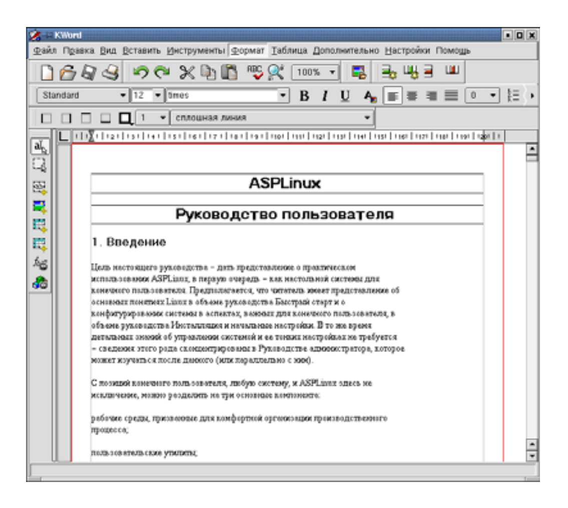


Рис. 11.37: Рабочее поле редактора **KWord**

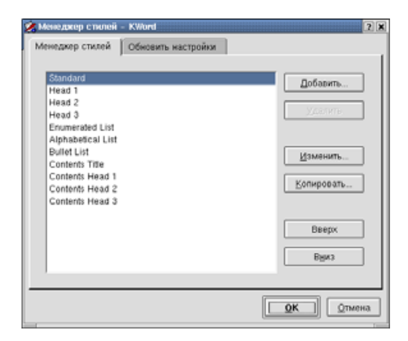


Рис. 11.38: Менеджер стилей **KWord**

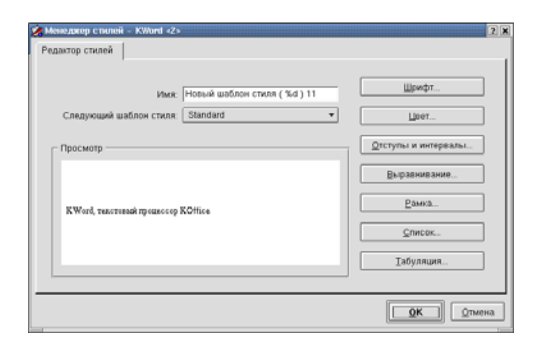


Рис. 11.39: Менеджер стилей — создание нового стиля

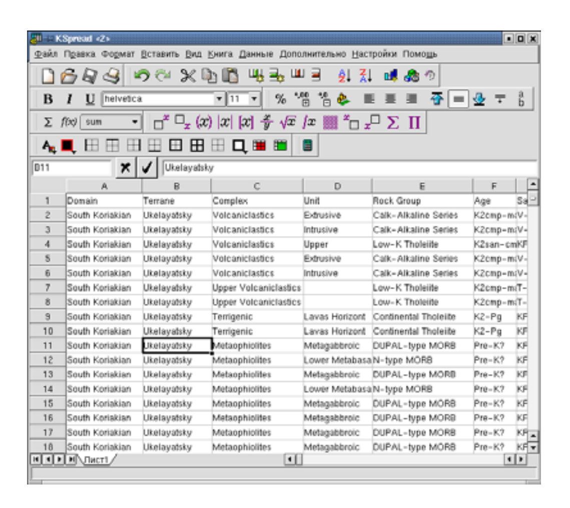


Рис. 11.40: Электронная таблица **KSpread** — внешний вид

Глава 12

Графика и мультимедиа

Дистрибутив **ASPLinux** включает в себя набор графических и мультимедийных приложений. Первые — это, с одной стороны, полнофункциональные средства для работы с растровой и векторной графикой, с другой — отдельные утилиты для выполнения частных задач.

Мультимедийные же приложения ориентированы в основном на любительские применения, связанные с воспроизведением аудио- и видеоинформации различных форматов. Это не значит, что в Linux невозможна профессиональная работа с мультимедиа данными — просто эти пакеты выходят за рамки ориентации дистрибутива.

12.1 GIMP — универсальная программа работы с растровой графикой

GIMP — универсальный полнофункциональный редактор растровой графики, единогласно признаваемый одним из самых удачных проектов, разрабатываемых в рамках модели открытых исходных текстов.

Запускается **GIMP** одноименной командой (`gimp`) из строки терминала или минитерминала, из стартового меню *KDE* (*«Графика»- «GIMP»*) или *GNOME* (*«Программы»- «Графика»- «GIMP»*). После этого загружается главная управляющая панель **GIMP** (рис. 12.1) и, возможно, еще две-три дополнительные панели, о которых будет сказано ниже.

Управляющая панель **GIMP** включает строку меню, серию инструментальных кнопок для выполнения различных действий, поля текущих цветов переднего и заднего плана и текущего инструмента (т.н. действующей кисти).

Главное меню содержит три пункта — *«Файл»*, *«Расширения»* и *«Справка»*. В первом пункте сосредоточены базовые операции (рис. 12.2):

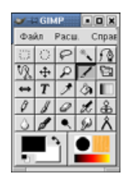


Рис. 12.1: Главная управляющая панель GIMP



Рис. 12.2: Главное меню управляющей панели — операции с файлами

- создание файла (*«Новый»*),
- открытие файла (*«Открыть»*),
- захват изображения с внешнего устройства, в том числе со сканера (XSane), и с экрана,
- *«Настройки»*,
- *«Диалоги»*,
- *«Выход»*.

Через меню *«Расширения»* доступны пункты (рис. 12.3.):

- *«Просмотр модулей»*,
- *«Детали дополнения»*,
- *«Просмотр базы данных»* дополнительных модулей,
- *«Редактор единиц измерения»*,
- *«Разложение видео на кадры»*,
- Доступ к пользовательским модулям (*«Скрипт-Фу»*).

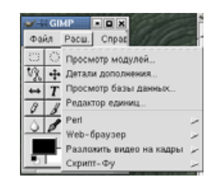


Рис. 12.3: Меню *«Расширения»* управляющей панели **GIMP**

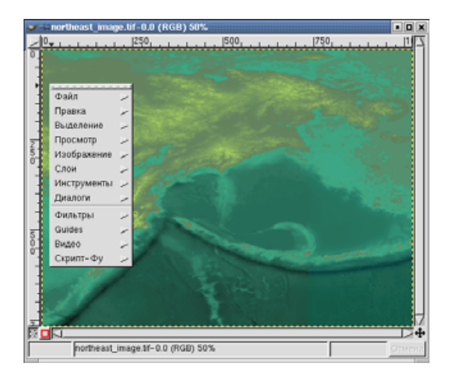


Рис. 12.4: Контекстное меню по правой клавише мыши на изображении

Здесь же сосредоточены ссылки на web-ресурсы по **GIMP** и средства программирования модулей на **Perl**.

Однако меню управляющей панели играет ограниченную роль в программе. Все основные манипуляции с изображением совершаются (после открытия или создания файла) щелчком правой клавиши мыши на нем. Этим вызывается всплывающее меню, содержащее многочисленные пункты (рис. 12.4).

Назначение первого пункта — манипуляции с файлами, а также общепрограммные настройки и печать. Следует отметить, что **GIMP** имеет собственный оригинальный формат файла — `XCF`. Именно в нем по умолчанию создается новый файл, для чего вызывается диалоговая панель, в которой определяются (рис. 12.5):

- размер изображения (высота и ширина в пикселях и дюймах); если в буфере присутствует скопированное изображение, создаваемый файл будет автоматически подстроен под его размер;
- горизонтальное и вертикальное разрешение (по умолчанию — 72 точки на дюйм);
- палитра создаваемого изображения — в RGB или градациях серого;

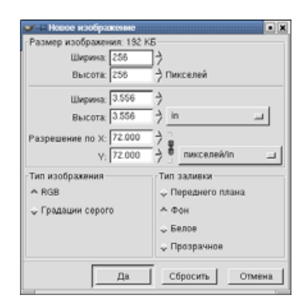


Рис. 12.5: Панель создания изображения

- тип заливки.

Кроме этого, **GIMP** позволяет напрямую открывать и редактировать файлы практически в любом распространенном растровом формате — TIFF, GIF, JPEG, PNG, BMP, включая формат Adobe PhotoShop, AVI, EPS, WMF и многие другие. И по умолчанию (*«Сохранить»*) сохраняет измененный файл в том же формате, в котором он был загружен.

Функция экспорта изображения осуществляется командой *«Файл»- «Сохранить как...»* Список экспортируемых форматов столь же широк, как и набор импортируемых типов файлов. При экспорте учитывается характер изображения: файл в индексированных цветах (например, gif) нельзя напрямую сохранить в полнолноцветном виде (как tif или jpeg). И наоборот, для сохранения tif-файла в формате gif необходимо предварительно выполнить индексацию.

Пункт *«Правка»* в основном аналогичен таковому стандартных Windows - программ (рис. 12.6): вырезание, копирование и вставка, отмена и повтор (число их уровней по умолчанию — 5). Возможно одновременное копирование нескольких изображений, для чего предназначены именованные буфера. Смысл остальных подпунктов обычно понятен из их названий (заливка, обводка и т. д.).

В пункте *«Выделение»* — всякого рода выделения (рис. 12.7): всего изображения, цветовых областей, плавающее, инвертирование выделения и запись выделенной области в виде канала.

В пункте *«Просмотр»* — изменение масштаба просмотра изображения (рис. 12.8): с шагом, кратным двум, точка за точкой или с помощью дополнительного навигационного окна (рис. 12.9), скрытие/показ линеек, направляющих и привязка к последним, а также открытие нового окна (с текущим файлом в нем).

Основные манипуляции с рисунком сосредоточены в пункте Изображение (рис. 12.10). Здесь его можно преобразовать (в RGB, индексированные цве-

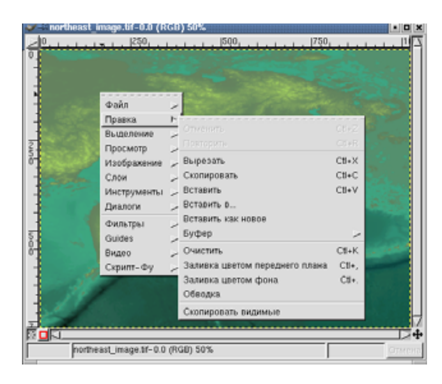


Рис. 12.6: Редактирование изображения — пункт *«Правка»* контекстного меню

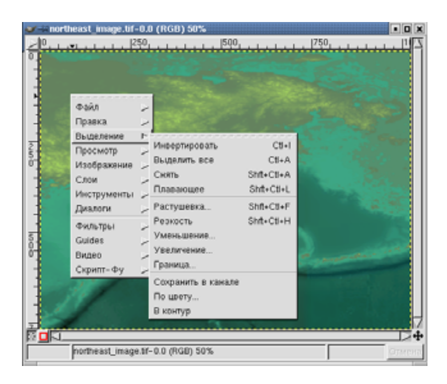


Рис. 12.7: Работа с выделенными областями через контекстное меню

та или градации серого), трансформировать, вращать, обрезать и т. д. Здесь же работа с альфа-каналами и цветовыми палитрами. С помощью подпунктов *«Масштабирование»* и *«Размер холста»* осуществляется изменение размеров изображения с масштабированием или без него, соответственно.

Смысл пункта *«Слои»* ясен из названия — в нем осуществляются различные операции с многослойным изображением. В пункте *«Инструменты»* — множество подпунктов (рис. 12.11): выделение областей различной формы (многоугольников, полигонов, произвольной формы и т. д.), перемещение, обрезание, зеркалирование, вращение (что частично совпадает с некоторыми подпунктами из пункта Изображение) и т.д. Из данного пункта также производится работа с текстом и рисование.

Следует заметить, что практически все действия пункта *«Инструменты»* до-

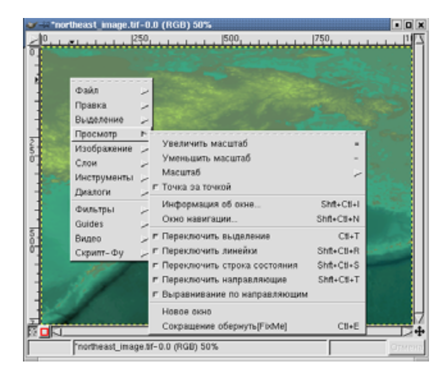


Рис. 12.8: Пункт *«Просмотр»* контекстного меню

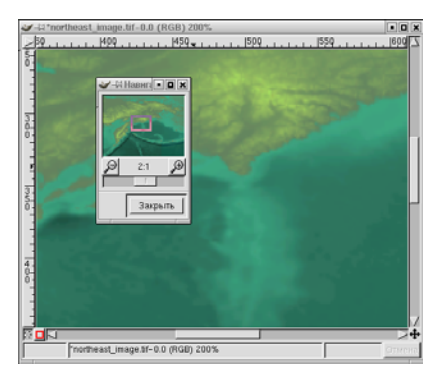


Рис. 12.9: Масштабирование изображения с помощью навигационного окна

ступны и через инструментальные кнопки главной управляющей панели.

Пункт *«Диалоги»* аналогичен таковому меню главной панели. Это вызов всякого рода дополнительных панелей: слоев, каналов, кистей, палитр, шаблонов, градиентных заливок и т.д.

Следующий пункт *«Фильтры»* обеспечивает наложение разнообразных спецэффектов. Иногда ряд пунктов неактивен. Обычно это значит, что данный фильтр просто неприменим к типу открытого файла. Так, например, гауссово размытие по определению нельзя применить к файлу формата GIF.

Самое интересное в **GIMP**'е — пункт *«Скрипт-Фу»*, не имеющий аналогов в других растровых редакторах. Это набор скриптов для создания пользовательских спецэффектов. Что в целом близко к назначению фильтров, но

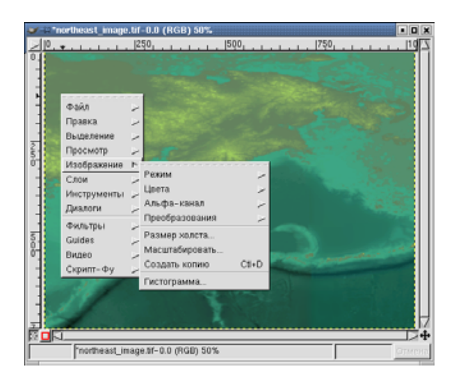


Рис. 12.10: Основные манипуляции с рисунком — пункт *«Изображение»* контекстного меню

отличается тем, что скрипты эти — набор интерпретируемых команд на языках (Tcl, Scheme и Perl), посредством которых можно создавать какие-нибудь уникальные эффекты, Кроме того, можно однократно запрограммировать какой-либо эффект и затем распространить его на различные исходные изображения. В отличие от применения фильтров, действие скриптов отменить (*«Отмена»*) как правило невозможно, т.к. скрипты представляют собой последовательность стандартных эффектов, количество которых может значительно превышать количество уровней отмены.

Кроме этого, в **GIMP** имеются функции для работы с видеопоследовательностями. Так, файл AVI (правда, только некомпрессированный) может быть разложен на составляющие его кадры, для которых можно изменить последовательность, применить сдвиг, удалить отдельные и т.д. Возможна и обратная процедура — сборка видеопоследовательности из отдельных изображений.

12.2 Прочие средства для работы с растровой графикой

Возможности, предоставляемые **GIMP**, часто оказываются излишними для простых манипуляций с изображениями (масштабирования, коррекции яркости/контрастности, преобразования из формата в формат). На этот случай в дистрибутив **ASPLinux** включен ряд более простых утилит для работы с растровой графикой.

К таким средствам относится универсальный пакет **ImageMagic**, представляющий собой набор отдельных утилит, часть из которых работает в командной оболочке, не требуя загрузки X Window System. Однако в сеансе последней доступен общий интерфейс для всех этих утилит, запускаемый командой

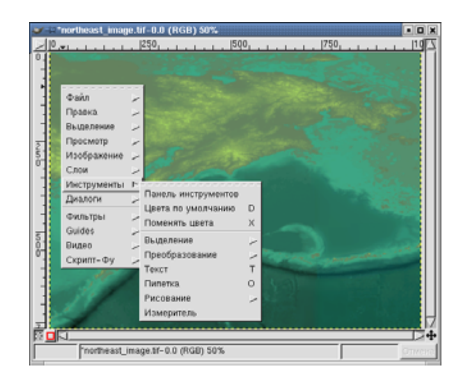


Рис. 12.11: Пункт *«Инструменты»* для преобразования изображений

`display` в окне терминала или строке минитерминала (рис. 12.12).

Утилиты пакета **ImageMagic** позволяют:

- монтировать ряд изображений в одно, выполняя функцию создания альбома в таких программах для Windows, как Ulead PhotoImpact;
- конвертировать в пакетном режиме любое количество файлов одного формата в другой (например, TIFF в GIF или JPEG);
- накладывать несколько изображений одно на другое.

Эффективные средства для просмотра серии изображений — программы **XV** (рис. 12.13) и **Electric Eyes** (рис. 12.14). Обе они обеспечивают одновременное открытие многих файлов (например, всех изображений из каталога) и навигацию по ним, в том числе и организацию слайд-шоу. Кроме того, в них входят средства несложного редактирования изображений и конвертации графических форматов (на данный момент эти средства просмотра не входят в поставку **ASPLinux**).

Для каталогизации и просмотра графических файлов предназначена программа **GQview** (рис. 12.15). Она не содержит встроенных средств редактирования изображений, но допускает подключение внешних редакторов в большом количестве (по умолчанию это **GIMP**, **Electric Eyes**, **XV** и **Paint**).

Наконец, в составе графической среды *GNOME* в качестве штатного средства имеется программа **gPhoto** (рис. 12.16). Изначально она предназначена для работы с цифровыми фотокамерами (список поддерживаемых моделей весьма обширен). Однако в ней есть довольно удобные средства каталогизации графических файлов. Возможно также несложное редактирование изображений — изменение размера, яркости, контрастности, цветового баланса,



Рис. 12.12: Интерфейс набора утилит **ImageMagic**

вращение и отражение. Наконец, средства экспорта в формат HTML позволяют легко создавать галереи изображений для представления в Интернет.

Из специализированных программ для захвата изображений следует отметить `ksnapshot`, предназначенную для снятия экранных копий (так называемых скриншотов). Она несколько напоминает `Capture` из комплекта CorelDraw, хотя и попроще (рис. 12.17).

Программа `ksnapshot` позволяет снять весь экран или только активное окно, скрыв или, напротив, выведя на первый план окно самого `ksnapshot`'а; в отличие от `Capture`, нельзя скопировать объект произвольной формы. Результаты можно сохранить в виде gif, jpeg, bmp и еще нескольких форматах, среди которых, правда, отсутствует tiff.

12.3 Средства для работы с векторной графикой

К сожалению, векторная графика не является сильной стороной платформы Linux: для нее не существует ничего функционально близкого современным векторным редакторам для Windows и MacOS. Исключения — рисовальный модуль из пакета **OpenOffice**, который по своим изобразительным возможностям примерно соответствует CorelDraw 5-й или 6-й версий. И при этом обладает некоторыми уникальными особенностями, например, возможностью создания истинно трехмерных изображений. Он подробно описан в руководстве по **OpenOffice**.

Кроме этого, для создания векторной графики предназначен графический модуль из интегрированного офисного пакета **KOffice** - **Kontour**. Он имеет базовый набор рисовальных средств, предусматривающий:

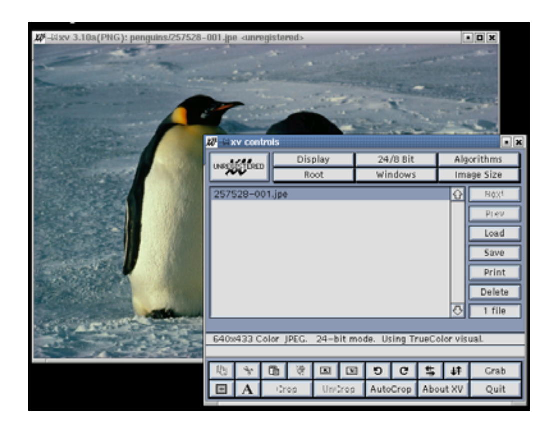


Рис. 12.13: Программа **XV** для просмотра и редактирования изображений

- рисование произвольных (т.н. `Freehand`) и ломаных линий (прямые рассматриваются как частный случай последних);
- непосредственное создание кривых Безье — в терминологии **Kontour** это не то же самое, что произвольные линии;
- создание прямоугольников и овалов (при нажатой клавише `Ctrl` - правильных квадратов и кругов), а также многоугольников.

Имеются средства ввода текста с возможностью изменения гарнитур, начертаний и кеглей. Текстовые элементы могут вращаться и деформироваться в горизонтальной и вертикальной плоскостях.

Имеется возможность точечного редактирования линий, в том числе и ограничивающих фигуры. Точки (узлы, в терминологии русской версии CorelDraw) могут быть перемещены, удалены или добавлены. Можно также разорвать линию, прямоугольник или овал. Для редактирования точек на второй линии (или фигуре) требуется предварительно перейти в режим указателя.

Для линий, фигур и текстовых элементов доступно изменение атрибутов (толщины, цвета и стиля линий, шрифтового оформления для текста, заливки - сплошной, градиентной или узорной (`Pattern`) для фигур. Оно осуществляется либо через контекстное меню по щелчку правой клавишей мыши (пункт *«Properties»*, вызывающий панель с закладками *«Info»*, *«Outline»*, *«Fill»*, либо через главное меню (*«Edit»*- *«Properties»* с тем же результатом).

Векторное изображение может сопровождаться растровой подложкой, определяемой через пункты меню *«Edit»*- *«Insert»*- *«Bitmap»*. Среди доступных форматов — GIF, JPEG, PNG, XMP.



Рис. 12.14: Программа **Electric Eyes** для просмотра и редактирования изображений

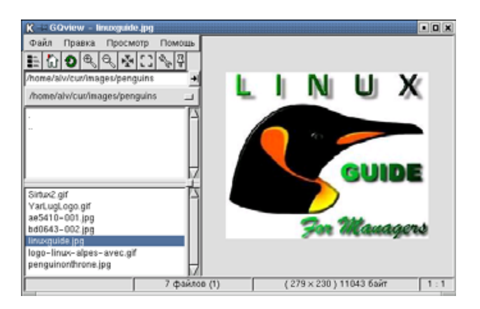


Рис. 12.15: Программа **GQview** для просмотра и каталогизации изображений

Возможности обмена данными ограничены. Импорт возможен только для изображений самого **Kontour** и документов **Xfig** (`*.fig`). Созданные изображения могут быть экспортированы в виде растра (в форматах GIF и XMP) или записаны как файл `Encapsulated PostScript` (`*.eps`) или `Scalable Vector Graphics` (`*.svg`).

12.4 Средства воспроизведения звука

Дистрибутив **ASPLinux** содержит ряд средств для воспроизведения звука различного рода и различных форматов. В первую очередь тут нужно упомянуть об инструментах для проигрывания стандартных дисков аудио-CD. Самый простой из них — консольная утилита `cdplay`, запускаемая из командной строки одноименной командой. При наличие в приводе аудио-CD начинается его

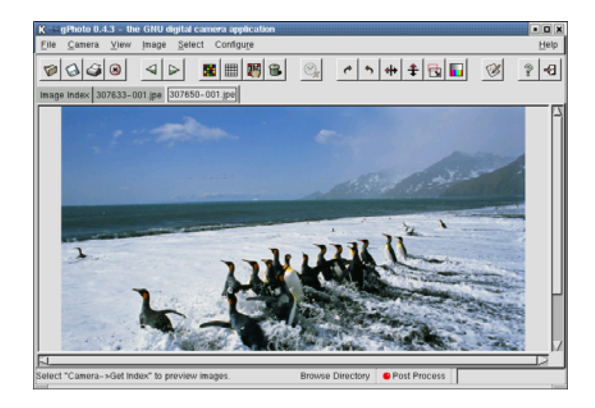


Рис. 12.16: Программа gPhoto для работы с растровыми изображениями и цифровыми камерами

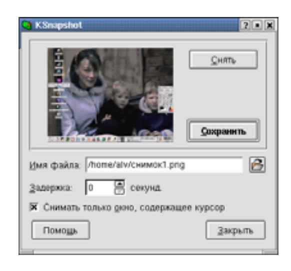


Рис. 12.17: Программа `ksnapshot` для снятия копий экрана

воспроизведение, продолжающееся до окончания записи, без возможности управления процессом. Однако программа немедленно (без ввода символа амперсанда) возвращает приглашение командной строки, не препятствуя дальнейшей работе.

Самое простое средство воспроизведения аудио в формате MPEG — утилита командной строки `mpg123`. В качестве аргументов этой команды используются имена mpeg-файлов. Если задать аргумент как путь до каталога с mpeg-файлами, например, как

```
mpg123 ~/mpeg_audio/*.mp3
```

все они будут проигрываться последовательно. Правда, никаких элементов управления воспроизведением в этой программе не имеется.

Полезным дополнением к консольным инструментам для воспроизведения

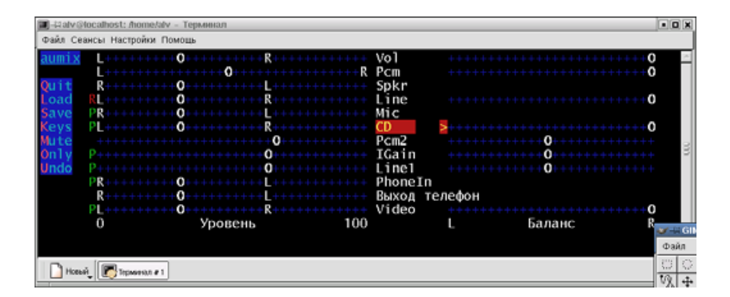


Рис. 12.18: Консольный микшер `aumix`

звука является программа `aumix`. Это простой микшер, запускаемый одноименной командой. Он позволяет (рис. 12.18) управлять воспроизведением CD-дисков, аудиофайлов формата mpeg, записью звука через микрофон и т.д.

Делается это двояко. Либо консольная программа для воспроизведения звука запускается в фоновом режиме, например:

```
mpg123 ~/mpeg_audio/*.mp3 &
```

после чего запускается программа `aumix`. Либо последнюю можно запустить на другой виртуальной консоли и управлять воспроизведением с нее.

Более развитые средства воспроизведения звука предназначены для работы в графическом режиме. Так, в комплекте каждой из интегрированных сред — *KDE* и *GNOME*, — имеется собственный CD-проигрыватель, доступный через соответствующие стартовые меню.

Существуют и средства проигрывания mpeg-файлов для графического режима.

Одним из лучших считается программа `xmms` (`X MultiMedia System`) (рис. 12.19). Эта программа позволяет открывать сразу все (или только выбранные) файлы каталога и, соответственно, добавлять их к плейлисту.

В набор дополнительных программ дистрибутива **ASPLinux** входит бесплатная версия программы **RealPlayer** для Linux, предназначенной для воспроизведения аудио- (и видео-) файлов соответствующего формата. Ни по интерфейсу, ни по возможностям она не отличается от Windows версии этой программы.

Для оцифровки звуковых дорожек аудио-CD предназначена универсальная программа grip (рис. 12.20). Она запускается из стартового меню *KDE* (*«Programms»- «Мультимедиа»- «grip»*) или *GNOME* (*«Программы»- «Мультимедиа»- «grip»*), а также их окна терминала или строки минитерминала одноименной командой).



Рис. 12.19: Программа **XMMS** для воспроизведения аудиофайлов формата mpeg

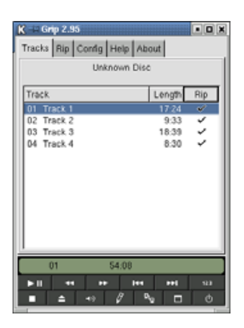


Рис. 12.20: Программа `grip` для оцифровки аудио-CD

После запуска программы (при вставленном в привод аудио-CD) в окне программы (закладка `Tracks`) появляется список всех дорожек диска. Двойной щелчок левой кнопки мыши на любой из них приводит к воспроизводству соответствующей дорожки. Для ее оцифровки дорожку следует отметить (щелчком правой кнопкой мыши); можно отметить несколько дорожек или весь диск (в последнем случае — щелчком правой кнопкой мыши на заголовке *«Rip»*, см. рис. 12.20).

Отметив требуемые дорожки, следует перейти на закладку *«Rip»* (рис. 12.21).

Здесь пользователю предоставляется две возможности оцифровки:

- запись дорожек в виде звукового файла в формате WAV (с помощью кнопки RIP ONLY);
- запись с одновременным кодированием в формат mpeg.

Для оцифровки дорожек и конвертации в mpeg grip использует внешние программы, которые определяются через закладку *«Config»* (рис. 12.22).

Созданные програмой `grip` аудиофайлы по умолчанию помещаются в подкаталог `~/mp3` в домашнем каталоге пользователя, где получают имена вида

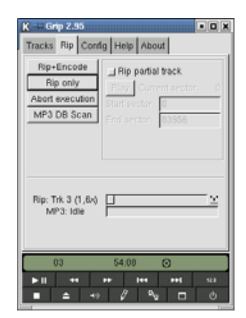


Рис. 12.21: Программа `grip`, закладка *«Rip»* для оцифровки аудиодорожек

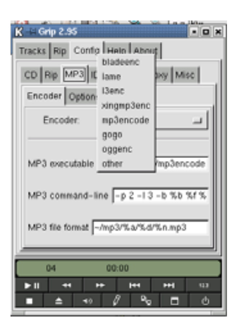


Рис. 12.22: Настройка программы `grip`

`track1.wav`, `track2.wav` и т.д. (или `track1.mp3` и т.д. для компрессированных файлов).

12.5 Средства воспроизведения видео

Текстовая программа `mtvp` имеет множество параметров, из которых практически используется лишь несколько из них. Так, для воспроизведения VideoCD достаточно (при вставленном в привод, но не смонтированном диске) набрать в командной строке терминала:

```
mtvp vcd:#
```

где # — номер трека, соответствующий порядковому номеру dat-файла на диске Video-CD.

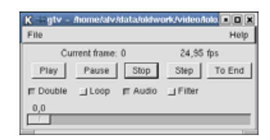


Рис. 12.23: Программа gtv для воспроизведения mpeg-видео

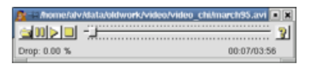


Рис. 12.24: Программа aviplay для воспроизведения видео

С помощью дополнительной опции -df видеофильм может быть запущен в полноэкранном режиме. Впрочем, это работает далеко не на всех видеокартах. Тем не менее качество видео вполне удовлетворительное.

Обобщенный формат запуска программы

```
mtvp {option} [URL]
```

поскольку, кроме видеодисков, она может воспроизводить mpeg-файлы с локального диска и из Сети — с http- или ftp-серверов. Полный список опций можно, как обычно, получить, введя в терминале

```
mtvp -h
```

Так, для определения глубины цвета используется опция d(n); если ее опустить — видео воспроизводится в системной палитре.

Еще одна программа для воспроизведения видео в формате MPEG - gtv, запускаемая из сеанса X Window System одноименной командой (в окне терминала или строке минитерминала). Она позволяет считать mpeg-файл (с CD или жесткого диска) и осуществлять управление его воспроизводством (рис. 12.23).

Наконец, для просмотра видео в формате AVI предназначена программа aviplay, запускаемая одноименной командой (рис. 12.24). Она проста в использовании и позволяет открывать avi-файлы и управлять их воспроизведением.

Следует учесть, что в оригинальном виде программа aviplay может воспроизводить только некомпрессированные avi-файлы. Для просмотра видеопоследовательностей, сжатых по какому-либо алгоритму, потребуются дополнительные программы-кодеки, ссылки на которые можно найти на сайте разработчиков aviplay.

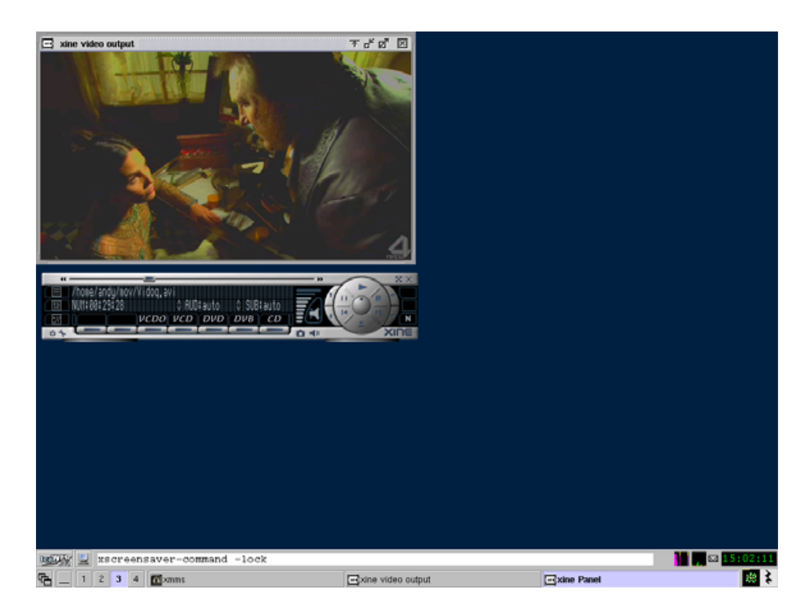


Рис. 12.25: **Xine** в действии

Не стоит на месте разработка более универсальных и мощных проигрывателей, таких как **Xine** (рис. 12.25).

После установки, чтобы не настраивать все вручную, желательно запустить утилиту `xine-check`, которая поможет автоматически установить параметры, необходимые для нормальной работы программы (сервер звука, видеодрайвер и т.д.). После этого уже можно запускать программу. Разработчики утверждают, что полная скорость передачи кадров для MPEG-2 будет достигнута на PII 400МГц.

Проигрыватель имеет модульную конструкцию, у него отделена как графическая оболочка от непосредственно программы воспроизведения, что позволяет использовать альтернативные (о которых ниже), так и любой новый кодек можно добавить без проблем — программа при каждом запуске проверяет их наличие. Модули программы могут быть оптимизированы под инструкции MMX, SSE и 3Dnow, а также реализована поддержка многопроцессорных систем. Как водится, интерфейс можно изменить с помощью обложек (рис. 12.26).

Проигрыватель работает со всеми мыслимыми и немыслимыми форматами файлов, а также с различными видео- и аудиокодеками. Дополнительно возможен запуск с различными аудио- (OSS, ALSA, aRts, ESD, Irix и Sun Audio)

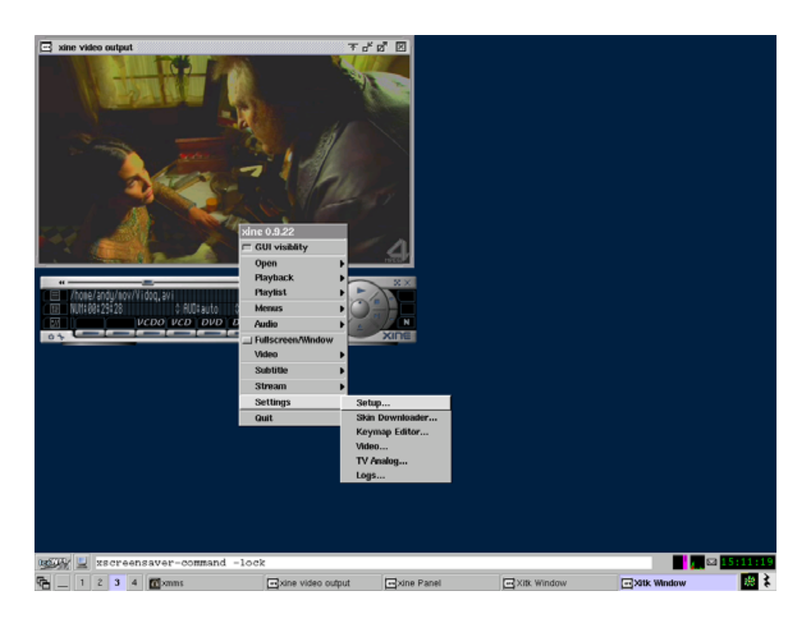


Рис. 12.26: Меню установок программы **Xine**

и видеодрайверами (Xvideo, XShm, OpenGL, SDL, ASCII Art library, Syncfb, framebuffer), применяемыми в Linux. Единственное с чем может не работать программа, так это с зашифрованными DVD-дисками. Это ограничение возникло по причине возможного юридического преследования.

При запуске **Xine** из командной строки можно передать ряд параметров.

- параметр `-A` позволяет задать используемый аудиодрайвер, иногда после запуска из-за его неправильной установки xine заканчивает работу, поэтому пекомендуется набрать `-A null`, а затем установить с помощью меню *«Option»* (рис. 12.27);
- аналогично предыдущему для видео есть параметр `-V`;
- параметр `-f` позволяет запустить воспроизведение сразу в полноэкранном режиме;
- спрятать графическую оболочку можно включив параметр `-d`;
- убрать рамку вокруг окна можно опцией `-B`;
- запустить воспроизведение сразу после активации можно опцией `-p`;

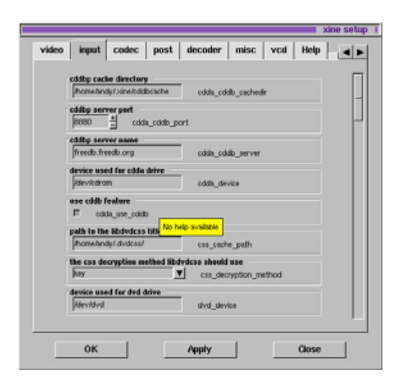


Рис. 12.27: Основное окно настроек

- параметр -G WxH[+X+Y] позволяет задать размер и положение окна просмотра (например, xine -G 800x600 установит размер окна равным 800 на 600 точек;

Все установленные значения затем автоматически записываются в конфигурационный файл в домашнем каталоге ~/.xine/config.

Дополнительно можно запустить проигрыватель с опцией -n, что даст возможность управлять xine через сеть. Для реализации этого необходимо создать файл ~/.xine/passwd и в нем добавить строки, разрешающие пользователям подключаться. В самом общем случае там может быть прописана строка ALL:ALLOW, разрешающая управление всем пользователям. Теперь в файл /etc/services добавьте строку

```
xinectl 6789/tcp # xine control
```

И, введя

```
telnet localhost 6789
```

можно подключиться к xine.

Об используемых командах можно узнать, набрав help [command] или syntax <command>.

Еще интересная опция MRL (media resource locator). Запустив с ней **Xine**, можно передать на него файл, набрав путь к нему в строке web-браузера, например, выбрав один из нижеперечисленных вариантов:

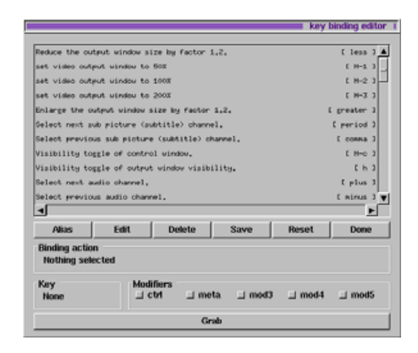


Рис. 12.28: Окно настройки привязки клавиш

```
file://<путь>
fifo://<путь>
stdin://mpeg2
tcp://<узел>:<порт>
http://<узел>
vcd://<номер_дорожки>
```

Таким образом предоставляется очень удобная возможность посмотреть видео из Интернета.

И конечно же **Xine** можно управлять с помощью комбинаций клавиш. Перезакрепить их можно в файле `~/.xine/keymap`, который имеет вполне понятную структуру. Необходимо только учитывать, что `meta` соответствует `Alt` на клавиатуре (рис. 12.28).

Наиболее из часто используемых приводятся ниже в таблице 12.1.

Кроме вышеперечисленных конфигурационных файлов, чтобы не задавать каждый раз аргументы в командной строке, можно использовать дополнительно файл `~/.xine/xinerc`, например:

```
--geometry 800x600+0+0
--network
--hide-gui
-p
```

Теперь при старте **Xine** программа запустится в окне `800x600`, включится сервер удаленного управления, автоматически включится воспроизведение видеофайла без графической оболочки.

В дополнение немного о различных графических оболчках к этой программе. Прежде всего стоит упомянуть **toxine** — скриптовая, командно-управляемая

Клавиша	действие
0...9	устанавливают просмотр на позицию 0%...90%
A	установить режим просмотра AUTO/16:9/4:3/DBV
Alt+c	вывести/спрятать окно настройки видео
F	переключение в широкоэкранный режим и обратно
G	вывести/спрятать графический интерфейс
H	как и предыдущая, но с окном воспроизведения
Ctrl+m	включение/выключение звука
t	снять snapshot
Enter	воспроизведение
Пробел	пауза
Up или Down	увеличение или уменьшение скорости воспроизведения
< или >	увеличить или уменьшить размер
Alt+1/2 или 3	установить размер окна воспроизведения равным 50%, 100% или 200%
n или m	позволяют догнать видео- или аудиопоток при асинхронном воспроизведении
Q	выход

Таблица 12.1: Основные клавиши интерфейса**Xine**

оболочка, которая позволяет в командной строке установить практически все параметры, доступные как из графической оболочки, так и скрытые от пользователя.

Другие оболочки основаны на библиотеке GTK. Сюда входят **gnome-xine**, довольно развитый **Sinek**, единственным недостатком которого является поддержка обложек, и последний — **Totem** работает только под GTK2.

Под управлением KDE удобнее будет воспользоваться услугами **kxine**, рассчитанного под Qt, в нем довольно простой интерфейс, не усугубляющий настройками.

С помощью модуля поддержки звукового сервера **aRts** можно заставить воспроизводить видео, используя проигрыватель **Noatun**. **Aaxine**, обеспечивающий вывод видео в виде ascii кода, входит в стандартную поставку **Xine** и будет доступен при компиляции с библиотекой aalib. В настоящее время ведутся разрботки модуля для web-браузера Mozilla. Как видите, **Xine** — довольно развитой видеопроигрыватель, имеющий множество опций и позволяющий с комфортом скоротать время за просмотром фильма.

Глава 13

Средства для работы в Интернет

13.1 Браузеры

Для ОС Linux существует множество браузеров.

Одним из наиболее развитых и эффективных является уже описанная выше программа konqueror. Однако традиционным средством web-серфинга на UNIX-платформах является текстовый браузер **Lynx**.

Lynx запускается в консольном режиме одноименной командой (`lynx`), с указанием URL требуемой страницы или без него. Для навигации по странице служат клавиши `Пробел` или `PageUp`/`PageDown`, навигация по ссылкам осуществляется клавишами управления курсором `Down` (вперед) и `Up` (назад).

Клавиши `Left` и `Right` действуют аналогично кнопкам **«Назад»** и **«Вперед»** в браузерах Windows .

Разумеется, **Lynx** не способен воспринимать достижения современных web-технологий. Однако он имеет свои преимущества. Во-первых, он исключительно быстр: если вы привыкли к **Netscape** или, тем более, к **Internet Explorer**, вас поразит скорость интерпретации html-кода. И при этом текстовая часть web-страницы будет вполне воспринимаема. Если, конечно, ее создатель спроектировал ее грамотно.

Во-вторых, **Lynx** очень строго подходит к интерпретации html-кода, требуя его соответствия спецификациям W3C. При серьезных нарушения последних он может просто отказаться читать страницу. И потому **Lynx** незаменим как одно из средств проверки валидности web-страниц.

Наконец, в третьих, при восприятии в основном текстовых web-материалов **Lynx** создает такое же ощущение зрительного комфорта, как и консольные текстовые редакторы для их набора. А со временем, возможно, вы увидите и своеобразную эстетику в его простоте и строгости…

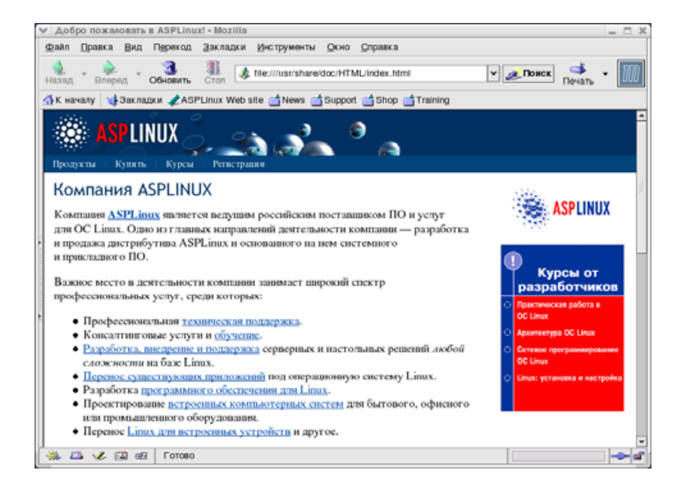


Рис. 13.1: Браузер **Mozilla**

О **Mozilla** сказать особенно нечего: это тот же самый **Mozilla** браузер что и для Windows как с точки зрения интерфейса, так и возможностей (рис. 13.1). Все навыки работы пригодятся и при использовании Linux-реализации.

Единственное существенное отличие — в комплекте с последней существенно меньше дополнительных подключаемых модулей (`plugins`).

Подобно своему Windows аналогу, **Mozilla** для Linux настраивается (в том числе и для работы с кириллицей) через пункты главного меню *«Правка»*-*«Настройки»* (рис. 13.2).

Сначала через пункт *«Отображение»* устанавливаются общие параметры коммуникатора, такие, как набор символов, шрифты и цвета для отображения различных частей документов.

Далее, в пункте *«Навигатор»*, можно установить стартовую страницу, которая заружается при запуске браузера, языки для представления документов (в частности, можно добавить русский язык), связать приложения с определенными дополнительными модулями.

13.2 Почтовые программы

В дистрибутив **ASPLinux** входит ряд консольных программ для получения и отправки почты, а также две программы графического режима — **Mozilla Mail** (компонент универсального пакета **Mozilla**) и **KMail**, входящий в интегриро-

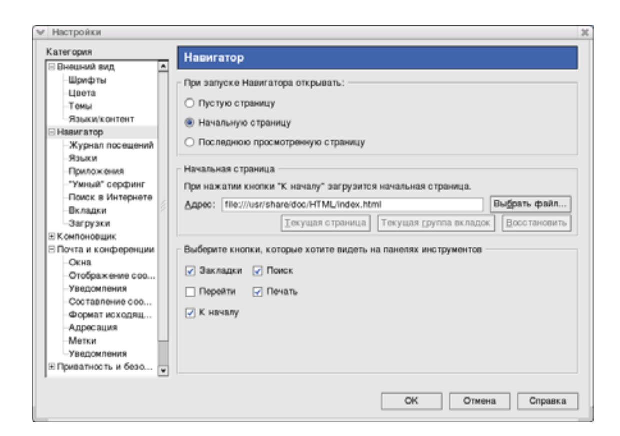


Рис. 13.2: Настройка браузера **Mozilla**

ванную среду *KDE*.

Mozilla Mail вызывается из главного меню **Mozilla** через пункты *«Окно»- «Почта и конференции»*. По интерфейсу и возможностям эта программ практически не отличается от своего Windows аналога (рис. 13.3).

Настройка **Mozilla Mail** осуществляется через пункт *«Правка»- «Параметры учетной записи»* самого почтового клиента **Mozilla Mail** (рис. 13.4). Здесь можно указать идентификацию пользователя (его полное имя, адрес электронной почты и т.д.), имена почтовых серверов POP и SMTP, каталог для хранения почты и прочие параметры.

Программа **KMail** несколько проще по внешнему виду, но, тем не менее, обеспечивает основные возможности работы с почтой. Запускается она из стартового меню *KDE* (*«Интернет»- «KMail»*) или из командной строки (командой kmail), после чего открывается окно программы (рис. 13.5).

Можно видеть, что основные действия по отправке и получению почты, составлению писем и управлению ими доступны как через главное меню, так и через панель инструментов. В последней предусмотрены кнопки для создания нового письма, сохранения и печати текущего, просмотра почтового ящика, ответа на письмо, пересылки и удаления его, доступа к адресной книге.

Имеется функция поиска писем (рис. 13.6). Он может осуществляться по теме писем (`subject`), имени отправителя и получателя, фрагментам в теле письма. Возможен также сцепленный поиск по двум полям (например, имени отправителя и фрагменту из тела).

Весьма многочислены настройки программы **KMail**, осуществляемые через пункты главного меню *«Настройки»- «Параметры»*. В вызываемой панели

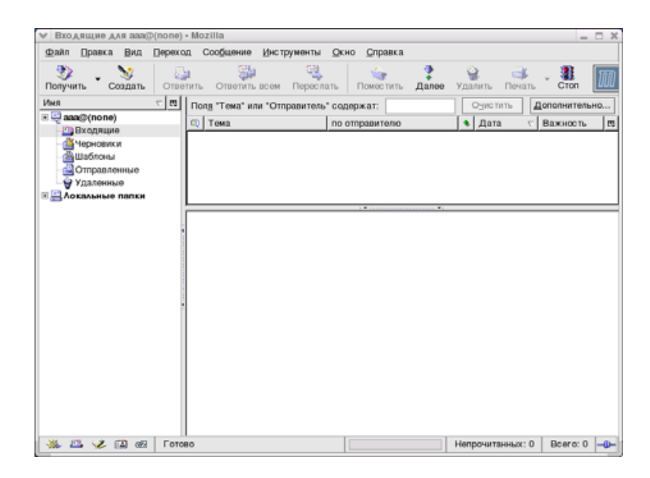


Рис. 13.3: Почтовая программа Mozilla Mail

(рис. 13.7), во-первых, можно определить несколько пользовательских профилей (с отдельным именем пользователя и адресом его электронной почты).

Далее, в пункте *«Сеть»*, настраиваются параметры входящего (Sendmail или SMTP) и исходящего почтовых серверов. Для последнего можно задать имя пользователя, его логин, пароль, имя сервера и его порт. Здесь же указываются условия получения почты (с удалением ее с сервера или без него), периодичность проверки почтового ящика, место сохранения полученной почты.

В пункте *«Внешний вид»* определяются шрифты и цветовая гамма для представления писем, формат представления списка писем, условия представления HTML. В пункте Редактор определяются язык и набор символов для составления и чтения почты, префиксы заголовков ответа, символы цитирования его.

13.3 Системы мгновенной передачи сообщений

Системы мгновенной передачи сообщений (в английской нотации — `Instant Messenger` или `IM`) работают по многочисленным протоколам. К наиболее известным из них следует отнести протоколы `MSN`, `Yahoo`, `AOL` (`OSCAR`), `Jabber` и `ICQ`.

Наиболее распространенным среди русскоязычного населения стали клиенты службы `ICQ`. Отчасти это объясняется хорошо изготовленным интерфейсом программы в 1995 году и поддержкой русского языка, который на тот момент, кроме `ICQ`, не поддерживал в достаточном объеме ни один клиент.

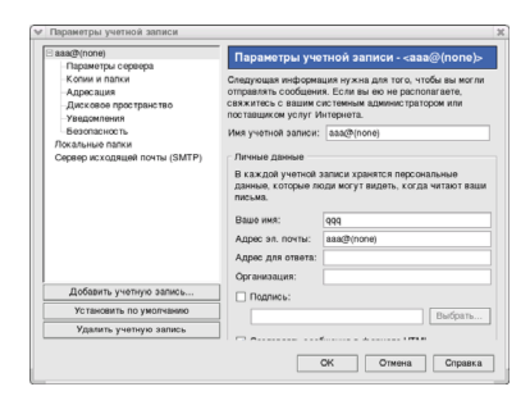


Рис. 13.4: Настройка почтовой программы Mozilla Mail

Для ОС Linux существует около десятка icq-клиентов. Наиболее популярные среди них — это **Sim**, **Licq** и **Gaim**, а также консольная версия такого клиента под названием **centerICQ**.

Клиент **Licq** (рис. 13.8) обладает весьма обширным набором возможностей. Для него существуют интерфейсы, базирующиеся на библиотеках `Qt` и `KDE`, последний из которых наиболее предпочтителен в силу более комфортабельной интеграции с рабочей средой. Также можно менять внешний вид и подключать массу модулей, благодаря которым можно получить ряд новых интересных возможностей, таких как проверка почтовых ящиков, удаленный доступ, пересылка сообщений на другой `UIN` или пейджер, вывод событий поверх всех окон (`On Screen Display`) и т.д. (рис. 13.9)

В основном окне настроек (рис. 13.10) доступно несколько вкладок, в каждой из которых в свою очередь множетсво различных опций. Самыми полезными могут оказаться опции хранения списка контактов на сервере (по умолчанию выключена), режим окна диалога — чат или пейджер, захват `URL` из буфера обмена и другие.

С версии `1.2.0a` появилась поддержка работы с списком контактов на сервере. Также следует обратить внимание, что **Licq** — один из немногих клиентов, которые поддерживают работу через прокси-сервер (рис. 13.11).

Данный снимок экрана не нуждается в особых пояснениях. Естественно, на прокси-сервере должны быть разрешены доступ к порту 443 и метод `CONNECT`. Поля «`Username/Password`» заполняются в случае, если прокси-сервер требует аутентификации.

Стоит заметить, что все остальные клиенты, работающие в графическом режиме, как правило имеют такой же минимум настроек, устанавливаемых подобным образом.

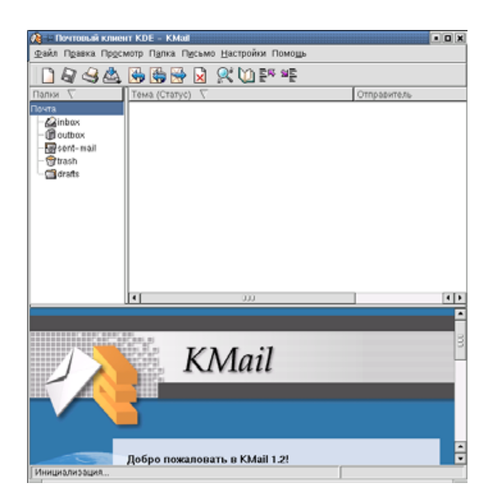


Рис. 13.5: Почтовая программа KMail

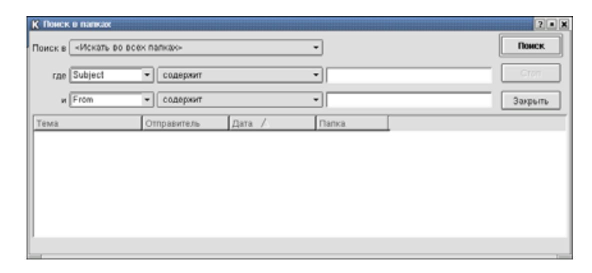


Рис. 13.6: KMail — поиск писем

В дополнение необходимо упомянуть менее распространенные клиенты, которые работают с протоколом ICQ. Среди них можно выделить:

- **gnomeICU**, поставляющийся в комплекте оконного менеджера Gnome;
- **micq** — простой текстовый клиент для Linux и Windows . Одной из его особенностей является возможность использовать сразу несколько приложений с разными UIN без специальных ухищрений;
- **eicq** порадует поклонников Emacs/Xemacs. Он написан на Emacs Lisp и обладает пока лишь минимальным набором возможностей, но, тем не менее, на него стоит обратить внимание.

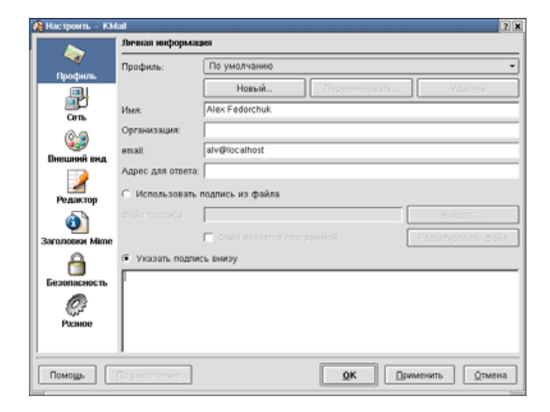


Рис. 13.7: Настройка программы KMail

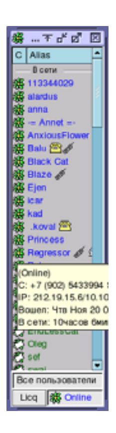


Рис. 13.8: Дружелюбный интерфейс **Licq**

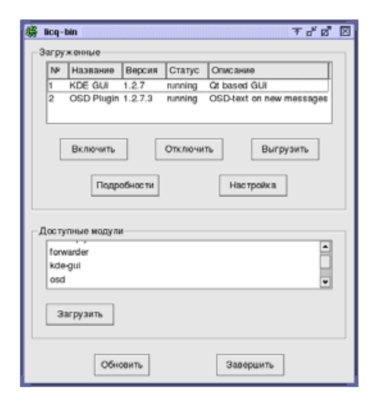


Рис. 13.9: Работа с модулями в **Licq**

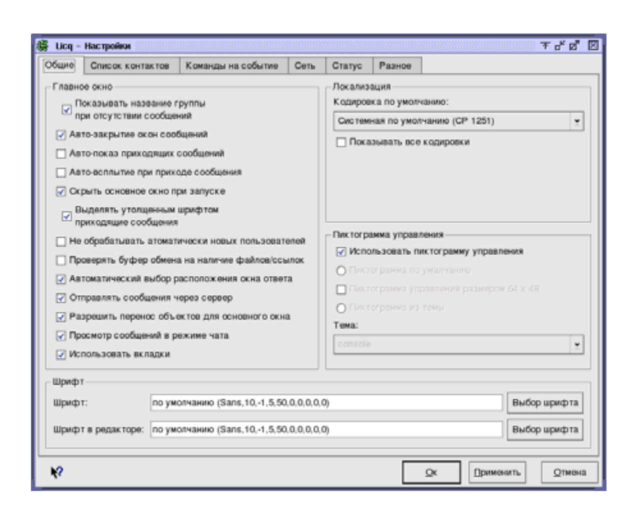


Рис. 13.10: Основное окно настроек

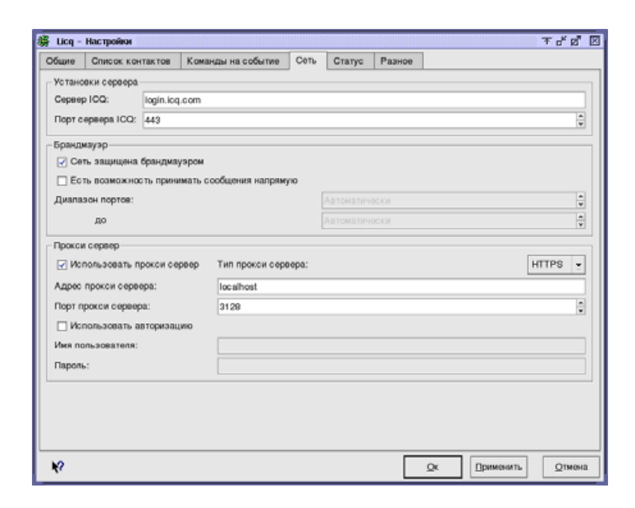


Рис. 13.11: Окно настройки прокси-сервера

Глава 14

Автоматическое обновление системы при помощи Yum

14.1 Общая информация

В состав дистрибутива **ASPLinux** входит **Yum** — система автоматической установки, удаления и обновления пакетов **RPM**. Она автоматически учитывает имеющиеся или возникающие в процессе установки или обновления пакетов зависимости и делает этот процесс прозрачным для пользователя.

Yum копирует заголовки из пакетов **RPM** с сервера (называемого репозиторием и представляющего собой HTTP-, ftp- сервер или, в случае если используется наша сборка **Yum**, просто место на диске) в свою рабочую область на клиентской машине. Когда требуется провести какое-то действие, то все процессы разрешения/выявления зависимостей и т.п. производятся непосредственно на клиентской машине, определяя, что же необходимо установить/удалить/обновить.

Yum поддерживает работу через прокси-сервер (сервер кэширования данных). Для использования прокси Вам необходимо настроить переменную окружения `http_proxy` на ваш прокси–сервер, например так:

```
http_proxy="http://www.someproxy.com:3128"
export http_proxy
```

После чего **Yum** сможет использовать Ваш прокси–сервер.

Для начала работы с **Yum** необходимо установить пакеты `yum`, `yum-headers` и `yum-repos`, если это не было сделано ранее в процессе установки дистрибутива.

14.1.1 Основные команды при работе с Yum

- Просмотреть список всех доступных к установке пакетов:
  `yum list`
- Этой команды достаточно для просмотра, иногда лучше воспользоваться командой:
  `yum list|less`
- Просмотреть только список обновлений:
  `yum list updates`
- Список установленных пакетов:
  `yum list installed`

В каждой из этих команд можно использовать дополнительный параметр: имя или шаблон имени пакета. Например:

`yum list kernel*`

покажет список доступных пакетов начинающихся с 'kernel'

Списки имеют унифицированный формат.

```
[root@simulakr root]# yum list kernel*
Получение информации о пакетах
Получение заголовков с сервера: ASPLinux Master Site
Получение заголовков с сервера: ASPLinux Updates
Получение заголовков с сервера: YUM-ASP Home repository
Кеширование информации
Поиск обновленных пакетов
Получение необходимых заголовков
Просмотр доступных пакетов
======================================================================
Name                           Arch          Ver-Rel
Size
----------------------------------------------------------------------
kernel-source                  i386       2.4.18-27.7asp    135.81 Мб
Просмотр установленных пакетов:
======================================================================
Name                           Arch          Ver-Rel
Size
----------------------------------------------------------------------
kernel                         i686       2.4.18-5asp        34.40 Мб
kernel-doc                     i386       2.4.18-5asp         5.16 Мб
kernel-pcmcia-cs               i386       3.1.27-18         828.32 Кб
kernel-source                  i386       2.4.18-5asp       130.68 Мб
```

Зачастую бывает нужно установить пакет не зная его имени, а имея в распоряжении только имя какого-нибудь файла, который должен принадлежать

этому пакету. Допустим, нам нужно найти, в каком пакете находится нужная нам библиотека:

```
[root@simulakr root]# yum provides /usr/lib/libcurl.so
Получение информации о пакетах
Получение заголовков с сервера: ASPLinux Master Site
Получение заголовков с сервера: ASPLinux Updates
Получение заголовков с сервера: YUM-ASP Home repository
Кеширование информации
Поиск обновленных пакетов
Получение необходимых заголовков
Поиск пакета предоставляющего: ['/usr/lib/libcurl.so']
 [100%] ################################ 0 найден среди доступных
 [100%] ################################ 1 найден среди установленых
Установленный пакет: curl-devel (curl-devel-7.9.7-1.asp.i386.rpm)
предоставляет '/usr/lib/libcurl.so'
```

Из чего видно, что пакет, содержащий такой файл, уже уставлен и называется curl-devel.

Не всегда есть возможность скачать и установить все что надо, поэтому прежде чем устанавливать новые пакеты, всегда можно прочитать информацию о них командой:

```
[root@simulakr root]# yum info mozilla
Получение информации о пакетах
Получение заголовков с сервера: ASPLinux Master Site
Получение заголовков с сервера: ASPLinux Updates
Получение заголовков с сервера: YUM-ASP Home repository
Кеширование информации
Поиск обновленных пакетов
Получение необходимых заголовков
Просмотр доступных пакетов
Просмотр установленных пакетов:
Name   : mozilla
Arch   : i386
Version: 1.2.1
Release: 0.2asp
Size   : 29.08 Мб
Group  : Приложения/Интернет
Summary: Браузер Web.
Description: Mozilla - это браузер www с открытым исходным кодом,
созданный по высоким стандартам производительности, скорости работы и
возможностью портирования.
```

14.1.2 Удаление, обновление и установка пакетов с помощью Yum

Существует три режима установки и обновления пакетов при помощи **Yum**

- Удаление пакетов
- Установка пакетов
- Обновление пакетов

Например, нужно удалить пакет php:

```
[root@simulakr root]# yum remove php
Получение информации о пакетах
Получение заголовков с сервера: ASPLinux Master Site
Получение заголовков с сервера: ASPLinux Updates
Получение заголовков с сервера: YUM-ASP Home repository
Кеширование информации
Поиск обновленных пакетов
Получение необходимых заголовков
Разрешение зависимостей
Зависимости разрешены
Необходимо выполнить следующее:
Удалено: php.i386
Разрешение зависимостей:
Удален: php-ldap.i386, php-pgsql.i386
Выполнить [y/N]: y
Рассчитывается доступное дисковое пространство.
Это может занять некоторое время.
Выполнено.
Старт транзакции RPM
warning: /etc/php.ini saved as /etc/php.ini.rpmsave
Удалено:  php.i386 php-ldap.i386 php-pgsql.i386
Транзакция Завершена
```

Yum выяснил, что для сохранения зависимостей нужно удалить еще и все зависимые от php пакеты php-ldap и php-pgsql.

Теперь php удален.

Установка пакетов — самый простой процесс. Пример:

```
[root@simulakr root]# yum install php-pgsql
Получение информации о пакетах
Получение заголовков с сервера: ASPLinux Master Site
Получение заголовков с сервера: ASPLinux Updates
Получение заголовков с сервера: YUM-ASP Home repository
Кеширование информации
Поиск обновленных пакетов
Получение необходимых заголовков
Разрешение зависимостей
Зависимости разрешены
Необходимо выполнить следующее:
Установка: php-pgsql.i386
Разрешение зависимостей:
```

```
Обновлено: php.i386 (not installed -> 0-4.1.2-7.3.6asp)
Выполнить [y/N]: y
Получение [updates] php-4.1.2-7.3.6asp.i386.rpm
Получение [updates] php-pgsql-4.1.2-7.3.6asp.i386.rpm
Рассчитывается доступное дисковое пространство.
Это может занять некоторое время.
Выполнено.
Старт транзакции RPM
 [100%] ################################# php
 [100%] ################################# php-pgsql
Установлено: php-pgsql.i386
Обновлено:  php.i386
Транзакция Завершена
```

Совершенно необязательно указывать весь список нужных пакетов. Нарпимер если указать только php-pgsql, то **Yum** сам определит, что необходимо установить также и php.

Для обновления пакетов используется команда yum update.

Рассмотрим её.

Допустим, я хочу обновить пакеты samba*:

```
[root@simulakr root]# yum update samba*
Получение информации о пакетах
Получение заголовков с сервера: ASPLinux Master Site
Получение заголовков с сервера: ASPLinux Updates
Получение заголовков с сервера: YUM-ASP Home repository
Кеширование информации
Поиск обновленных пакетов
Получение необходимых заголовков
Разрешение зависимостей
Зависимости разрешены
Необходимо выполнить следующее:
Обновлено: samba-common.i386 (0-2.2.6-1asp -> 0-2.2.7a-5.7.3asp),
samba-client.i386 (0-2.2.6-1asp -> 0-2.2.7a-5.7.3asp)
Выполнить [y/N]: y
Получение [updates] samba-client-2.2.7a-5.7.3asp.i386.rpm
Получение [updates] samba-common-2.2.7a-5.7.3asp.i386.rpm
Рассчитывается доступное дисковое пространство.
Это может занять некоторое время.
Выполнено.
Старт транзакции RPM
 [100%] ################################# samba-common
 [100%] ################################# samba-client
Обновлено: samba-common.i386 samba-client.i386
Транзакция Завершена
```

Для обновления всего дистрибутива нужно воспользоваться командой yum update без указания последующих параметров.

14.1.3 Настройка репозитория

Разложите ваши **RPM**-пакеты внутри одного каталога. Ничего страшного, если в нем будут подкаталоги, поскольку обычно администраторы стараются держать `i386` и `noarch` пакеты в разных подкаталогах.

Чтобы построить репозиторий, воспользуйтесь командой

`yum-arch -z имя_каталога_репозитория`

Добавьте запись о репозитории в файле, созданном в каталоге `/etc/yum.d` на клиентской машине. Например, если вы создали репозиторий в каталоге `/var/my-repository`, то в файл настроек `/etc/yum.d/mylocal` надо вписать следующие строки:

```
[mylocal]

name=Мой собственный репозиторий
baseurl=file:///var/my-repository
```

Глава 15

Заключение

В рамках настоящего руководства невозможно осветить все вопросы практического использования Linux. И потому пользователю, столкнувшемуся с ограниченностью своих знаний, неизбежно придется обращаться к дополнительным источникам информации. Для облегчения этой задачи и предназначен приводимый в заключении их обзор.

Наиболее полным руководством по принципам работы в командных оболочках может служить книга Б.Кернигана и Р. Пайка «UNIX — универсальная среда программирования» (М.: Финансы и статистика, 1992, с. 304). Сравнительную характеристику различных командных оболочек можно найти в книге А.Шевеля «Linux. Обработка текстов» (СПб: Питер, 2001, с. 384). В качестве руководства по программированию сценариев командной оболочки Shell рекомендуется работа: Соловьев А. Программирование на shell на сайте http://unixes.newmail.ru/unix1.htm.

Описания практически всех существующих оконных менеджеров и интегрированных сред, упомянутых в настоящем руководстве, содержатся на сайте http://www.plig.org/xwinman/index.html. Ряд оконных менеджеров, не рассмотренных в настоящей работе, описан в серии статей Виктора Вислобокова на http://t37.nevod.perm.su/linux/documents/.

Наиболее подробные описания прикладных пакетов под Linux можно найти в книгах Кая Петцке «Linux. От понимания к применению» (М.: ДМК, 2000, с. 576) и А.Федорчука «Офис, графика, Web в Linux» (СПб: БХВ-Петербург, 2001, с. 416).

Кроме этого, на многих русскоязычных сайтах содержатся полезные сведения об офисных пакетах (например, http://ppg.ice.ru/), графических редакторах (http://gimp.linux.ru.net/) и т.д.